Gnash Manual V0.3

Rob Savoye <rob@senecass.com>

This manual describes version 0.7.2 of Gnash.

Revision History

Revision Gnash Manual V0.3	Oct 2006

Rob Savoye <rob@senecass.com> Updated for the alpha release.

Free Software Foundation

Table of Contents

Introduction

Gnash is a free Flash movie player, which can be used standalone or as a Firebox/Mozilla plugin.

Gnash Overview

Gnash is originally based on the code of GameSWF, by Thatcher Ulrich. GameSWF was the most advanced of the free flash movie player projects, and implemented a fairly broad set of Flash Format v7 compliance. GameSWF was unsupported public domain software however, and not really designed as an industrial strength project which could be used by everyone who uses Firefox. So in early December of 2005, GameSWF was forked. After being rearranged in GNU project style, development of the plugin was started.

Gnash works as standalone player; as a Mozilla Firefox plugin; and as a Konqueror plugin, called Klash. The plugins use the standalone player for processing, but embed the output into the browser.

Currently, Gnash uses OpenGL to get a high quality rendered image. There are plans to write a Cairo back-end to give Gnash a performance boost on graphics processors without 3D hardware acceleration. Other plans involve running Gnash in a framebuffer, with no windowing system at all.

Included in the Gnash is an XML-based messaging system, as specified in the Flash specification. This lets a flash movie communicate over a TCP/IP socket, and parse the incoming XML message. This allows a movie be a remote control for other devices or applications.

Plugin Overview

The plugin currently works by creating a new window in the standalone player which is connected to the browser window in Firefox.

All movies are downloaded to /tmp and played from there. Many web pages use IE-specific JavaScript to initiate movies, which means that Firefox does not load the Gnash plugin.

Standalone Player Overview

The standalone player supports both SDL and GTK2. The SDL support is more portable, and the GTK support allows better integration as a Firefox plugin. When using GTK, a right-button mouse click will activate a menu which can be used to control the movie.

Flash Support:

Many movies play without any difficulty in Gnash. Gnash supports the majority of Flash opcodes up to SWF version 7, and a wide sampling of ActionScript classes for SWF version 7. All the core ones are implemented, and many of the newer ones work, but may be missing some of the methods. All unimplemented opcodes and ActionScript classes and methods print a warning when using -v with gnash or gprocessor. Using gprocessor -v is a quick way to see why a movie isn't playing correctly.

There are plans to work towards supporting all the SWF version 8 and greater opcodes, as well as as implementing the missing methods and ActionScript classes. During the first few months of Gnash's existence as a project, most of the focus has been towards portability issues, and getting the plugin to work. Now that the plugin works, more focus will be spent on catching up to full compliance with version 7

and beyond.

Currently implemented ActionScript classes are: Array, Boolean, Date, Key, Math, Mouse, MovieClip, Number, Object, Sound, String, XML, XMLNode, and XMLSocket.

Partially implemented classes are: MovieClipLoader, NetConnection, LocalConnection, TextField, and TextFormat.

Unimplemented classes are: Accessibility, Error, Function, LoadVars, Microphone, NetStream, Selection, SharedObject, Stage, System, Button, Camera, Color, ContextMenu, CustomActions, and Video.

Unimplemented Opcodes are: Throw, Implements, Extends, enum_object, Try, new_method, enum_object, md length, md substring, md chr, delete, and get target.

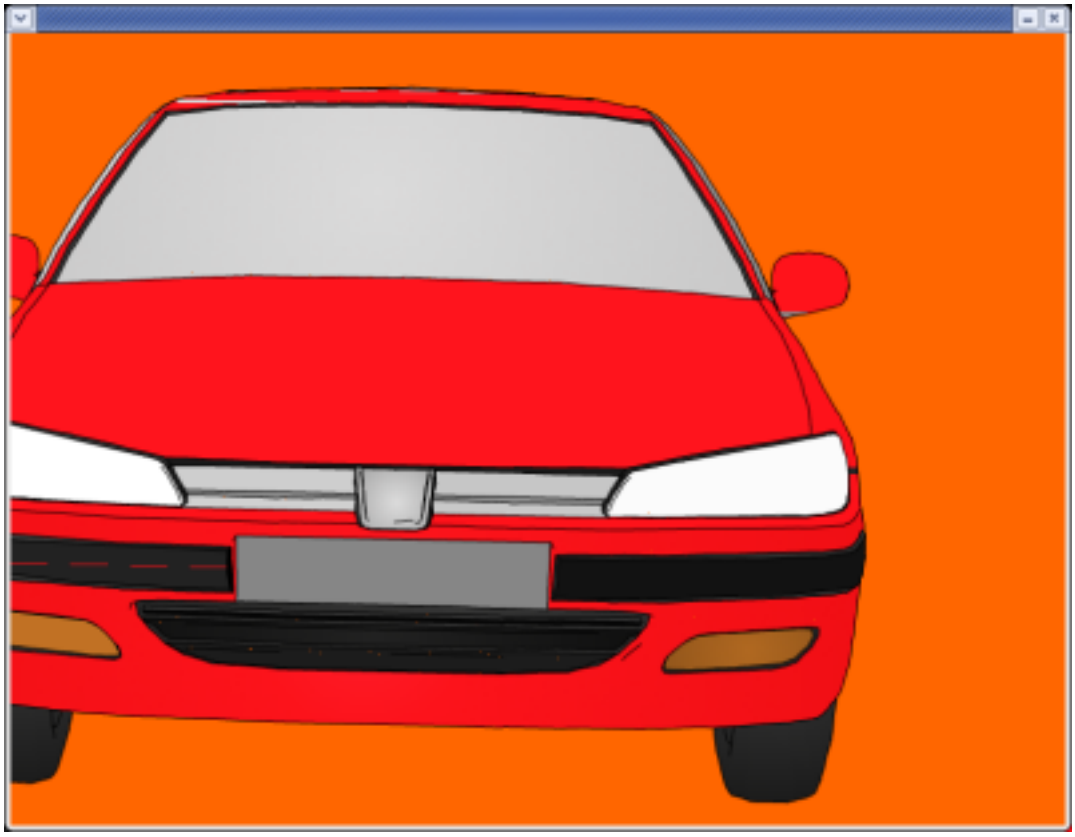
There is currently no support for FLV video, more than minimal AMF data, or loading external jpegs.

Gnash Usage

Currently only the standalone player is fully operational. You can execute any flash movie from the command line by just supplying the file name. No command line options are required to just play the movie using the default actions. So if you type:

gnash samples/car_smash.swf

it'll create a window and play the movie. In this case it's a simple animation of a car driving, swerving, and finally crashing.



Gnash Command Line Options

While by default no options are necessary, there are options that can be used to change *Gnash's* basic behavior.

gnash [options] file

-h	-h Print usage information.
-s factor	-s factor Scale the movie up/down by the specified factor.
-c	-c Produce a core file instead of letting SDL trap it. By default, SDL traps all signals, but sometimes a core file is desired to assist with debugging.
-d num	-d num Number of milliseconds to delay in main loop. The main loop polls continuously with a delay to adjust how long *gnash* sleeps between iterations of the loop. The smaller the number, the higher the CPU load gets, and of course, the more iterations of the main command loop.
-p	-p Run full speed (no sleep) and log frame rate.
-a	-a Turn anti-aliasing on/off. (obsolete)
-v	-v Be verbose; i.e. print debug messages to stdout.
-va	-va Be verbose about movie Actions.
-vp	-vp Be verbose about parsing the movie. Warning: this can generate a lot of text, and can affect the performance of the movie you are playing.
-ml bias	-ml bias Specify the texture LOD bias (float, default is -1) This affects the fuzziness of small objects, especially small text.
-w	-w Write a debug log called gnash-dbg.log. This will record of all the debug messages whether they are printed to the screen or not.
-j	-j Specify the width of the window. This is mostly used only by the plugin.
-k	-k Specify the height of the window. This is mostly used only by the plugin.
-1	-1 Play once; exit when/if movie reaches the last frame. This is the default.
-r [0|1|2|3]	-r [0|1|2|3] 0 disables rendering and sound (good for batch tests). 1 enables rendering and disables sound (default setting).

2 enables sound and disables rendering.

3 enables rendering and sound.

-t sec
-t sec
Timeout and exit after the specified number of seconds. This is useful for movies which repeat themselves.

-x id
-x id
This option is only used by Mozplugger when running the standalone player from within a web browser like Firefox. This specifies the X11 window ID to display in.

-b bits
-b bits
Bit depth of output window (16 or 32, default is 16).

-u url
-u url
Set the _url member of the root movie. This is useful when you download a movie and play it from a different location. See also the -U switch.

-U baseurl
-U baseurl
Set base url for this run. URLs are resolved relative to this base. If omitted defaults to the _url member of the top-level movie (see the -u switch).

-P parameter
-P parameter
Parameters are given in ParamName=Value syntax and are mostly useful to the plugin to honour EMBED tags attributes or explicit OBJECT PARAM tags. A common use for -P is to provide FlashVars (ie: -P "FlashVars=home=http://www.gnu.org").

Gnash Interactive Control Keys

While a movie is playing, there are several control keys. These can be used to step through frames, pause the playing, and control other actions.

CTRL-Q
CTRL-Q
Quit/Exit.

CTRL-W
CTRL-W
Quit/Exit.

ESC
ESC
Quit/Exit.

CTRL-P
CTRL-P
Toggle Pause.

CTRL-R
CTRL-R
Restart the movie.

CTRL-[or kp-
CTRL-[or kp-
Step back one frame.

CTRL-] or kp+
CTRL-] or kp+
Step forward one frame.

CTRL-A	CTRL-A Toggle anti-aliasing (doesn't work).
CTRL-T	CTRL-T Debug. Test the set_variable() function.
CTRL-G	CTRL-G Debug. Test the get_variable() function.
CTRL-M	CTRL-M Debug. Test the call_method() function.
CTRL-B	CTRL-B Toggle the background color.

User Configuration File

Gnash supports a configuration file which lives in the users home directory. This file is called *.gnashrc*. In this you can have default settings which will be used by Gnash when running standalone, or as a browser plugin. Any command line options override these values.

Gnash supports three types of configuration variables. The three types are an on/off value, a numeric value, or in the case of the whitelist and blacklist, a list of hostnames as ASCII text.

localdomain	localdomain This value can be set to either *on* or *off*, and controls the loading of external Flash movies over a network. Traditionally this tells Gnash to only load Flash movies from the existing domain.
localhost	localhost This value can be set to either *on* or *off*, and controls the loading of external Flash movies over a network. This is a stricter version of the *localdomain* setting as this allows the loading of Flash movies to the same host Gnash is running on.
whitelist	whitelist This is a list of hostnames, seperated by a colon *:*. If this list is not empty, only external flash movies from these hosts are allowed to load.
blacklist	blacklist This is a list of hostnames, seperated by a colon *:*. External flash movies from these domains are never allowed to load. If whitelist is present and not empty, blacklist is not used.
delay	delay Gnash uses a timer based event mechanism to advance frames at a steady rate. This lets one override the default setting in Gnash to play a movie slower or faster.
verbosity	verbosity This is a numeric value which defines the default level of verbosity from the player.
debuglog	debuglog This is the full path and name of debug logfile as produced by Gnash.

writelog	writelog This value can be set to either *on* or *off*, and controls whether a debug log is always written by Gnash, or not at all.
sound	sound This value can be set to either *on* or *off*, and controls the sound of the standalone player. By default Gnash enables playing the sound in any Flash movie.
pluginsound	pluginsound This value can be set to either *on* or *off*, and controls the sound of the player when running as a browser plugin. By default, sound is enabled when using Gnash as a browser plugin.

My current Gnash configuration file looks like this:

```
#
# Gnash client options
#

# Only access remote content from our local domain
set localdomain on

# Only access content from our local host
set localhost on

# These sites are OK
# uncommenting the following line will allow load of external
# movies *only* from the specified hosts.
#set whitelist www.doonesbury.com:www.cnn.com:www.9news.com

# Don't access content from these sites
set blacklist www.doubleclick.com:mochibot.com

# The delay between timer interupts
set delay 50

# The default verbosity level
set verbosity 1

# The full path to the debug log
set debuglog ~/gnash-dbg.log

# Write a debug log to disk
set writelog on

# Enable or Disable sound for the standalone player
set sound on

# Enable or Disable sound for the standalone player
set pluginsound on
```

Building From Source

Code Dependancies

Gnash has dependencies on other packages. When installing from a packaged release file (rpm, deb, etc.), you'll need to install the development versions to get the tools used to compile Gnash. The normal runtime packages installed are usually missing the headers needed to compile Gnash.

OpenGL

Gnash uses OpenGL for rendering the images. OpenGL is a 3D graphics package which supports hardware acceleration. You can get the free version of OpenGL at this link: http://www.mesa3d.org

To install a binary package using apt-get (on Debian based systems), install libgl1-mesa-dev. For RPM or Yum based systems, install the libmesa-devel package.

AGG

AGG is the AntiGrain low-level 2D graphics library used instead of OpenGL on embedded systems. This can be used on the desktop as well, but its primary purpose is to run without OpenGL.

To install a binary package using apt-get (on Debian based systems), install libagg-dev. For RPM or Yum based systems, install the agg-devel package.

GtkGlExt

GtkGlExt is an optional package used instead of SDL. Gtk enables better integration with Firefox, as well as better event handling and higher level GUI constructs like menus and dialog boxes.

To install a binary package using apt-get (on Debian based systems), install gtkglext-dev. For RPM or Yum based systems, install the gtkglext-devel package.

Pango

Pango is a dependency of GtkGlExt, and is used for font handling.

To install a binary package using apt-get (on Debian based systems), install pango-dev. For RPM or Yum based systems, install the pango-devel package.

Atk

Atk is a dependency of GtkGlExt, and is used for accessibility support.

To install a binary package using apt-get (on Debian based systems), install atk-dev. For RPM or Yum based systems, install the atk-devel package.

Cairo

Cairo is a dependency of GtkGlExt, and is used for 2D rendering.

To install a binary package using apt-get (on Debian based systems), install cairo-dev. For RPM or Yum based systems, install the cairo-devel package.

Boost

Boost is a library of portable C++ classes and templates which layer on top of STL. Boost is used for thread and mutext handling.

To install a binary package using apt-get (on Debian based systems), install boost-dev. For RPM or Yum based systems, install the libboost-devel package.

Glib

Glib is a dependency of GtkGlExt, and is a collection of commonly used functions.

To install a binary package using apt-get (on Debian based systems), install glib-dev. For RPM or Yum based systems, install the glib-devel package.

Gstreamer

Gstreamer is used for sound and video support. It is not needed to build this release. Currently only Gstreamer version 0.10 or higher can be used.

To install a binary package using apt-get (on Debian based systems), install streamer-dev. For RPM or Yum based systems, install the gstreamer-devel package. Version 0.10 or greater will be required.

FFMPEG

FFMPEG can also be used for sound and video support. It is not needed to build this release, but is recommended if you want working sound.

To install a binary package using apt-get (on Debian based systems), install ffmpeg-dev. For RPM or Yum based systems, install the libffmpeg-devel package. Version 0.10 or greater will be required.

SDL

The Simple DirectMedia Layer is a cross-platform multimedia library designed to provide low level access to audio, keyboard, mouse, joystick, 3D hardware via OpenGL, and 2D video framebuffer. You can get SDL from this link: http://www.libsdl.org [http://www.libsdl.org]

To install a binary package using apt-get (on Debian based systems), install libsdl1.2-dev. For RPM or Yum based systems, install the SDL-devel package.

PNG

PNG [http://www.libpng.org/pub/png/] is a patent-free image format that is comparable to *GIF*.

To install a binary package using apt-get (on Debian based systems), install libpng12-dev. For RPM or Yum based systems, install the libpng package.

JPEG

JPEG [http://www.ijg.org/]is a lossy image format, heavily used for images because of the smaller size of the file.

To install a binary package using apt-get (on Debian based systems), install libjpeg62-dev. For RPM or Yum based systems, install the libjpeg package.

libxml2

Libxml2 is the GNOME XML parser library. This is used when Gnash is configured with XML support. Libxml2 is used to parse any incoming messages when using the XML or XMLSocket ActionScript classes. You can get libxml2 from this link: http://xmlsoft.org

To install a binary package using apt-get (on Debian based systems), install libxml2-dev. For RPM or Yum based systems, install the libxml2-devel package.

Ogg Vorbis

Ogg Vorbis [http://www.vorbis.com/] is a patent free audio format that is comparable (many people say better) to MP3. This is used by SDL to play Ogg files. You can get Ogg Vorbis from this link: http://www.vorbis.com/ [http://www.vorbis.com/].

To install a binary package using apt-get (on Debian based systems), install libogg-dev. For RPM or Yum based systems, install the libogg package.

libMAD

libMad is a high-quality MPG decoder for audio files. All variations of the MP3 format are supported. http://www.underbit.com/products/mad/ [http://www.underbit.com/products/mad/]. You can get libMAD from this link: http://xmlsoft.org

To install a binary package using apt-get (on Debian based systems), install libmad0-dev. For RPM or Yum based systems, install the libmad package.

Mozilla/Firefox

The Mozilla development package is no longer needed to build the plugin. The required header files are now included in Gnash, so it builds without Mozilla or Firefox installed at all.

To install a binary package using apt-get (on Debian based systems), install mozilla-dev or firefox-dev. For RPM or Yum based systems, install the mozilla-devel or firefox-devel package.

Documentation Dependancies

Docbook

Docbook [http://docbook.sourceforge.net/] is an industry standard XML format for technical documentation. It is used by many projects, as there are free software implementations of the Docbook style-sheets and tools. It is used by both the GNOME [http://www.gnome.org] project, and the Linux Documentation Project. [http://www.tldp.org/]. It is customizable by using style-sheets for each output device. Default style-sheets are included for a variety of print formats, like *PDF* and *HTML*.

You can get Docbook from this link: http://sourceforge.net/project/showfiles.php?group_id=21935#files [http://sourceforge.net/project/showfiles.php?group_id=21935#files].

To install a binary packages using apt-get (on Debian based systems), install the docbook, docbook-xsl, docbook-xml, docbook-dsssl,and docbook-utils packages. For RPM or Yum based systems, install the docbook, docbook-style-xsl, docbook-style-dsssl, and docbook-utils packages.

DocBook2X

DocBook2X is a software package that converts DocBook documents into the traditional Unix man page format and the GNU Texinfo format. It supports tables for man pages, internationalization, and easy customization of the output using XSLT. This package is used to convert the Gnash documentation into HTML and Texinfo [http://www.gnu.org/software/texinfo/]formats. Texinfo can later be converted to standard GNU *info* pages.

You can get DocBook2X from this link: http://docbook2x.sourceforge.net/ [http://docbook2x.sourceforge.net/]. Texinfo is available at this link: http://ftp.gnu.org/gnu/texinfo/

[http://ftp.gnu.org/gnu/texinfo/].

To install a binary package of DocBook2X using apt-get (on Debian based systems), install docbook2x. For RPM or Yum based systems, install the docbook2x package. To install a binary package of DocBook2X using apt-get (on Debian based systems), install texinfo. For RPM or Yum based systems, install the texinfo package.

FOP

FOP (Formatting Objects Processor) is the world's first print formatter driven by XSL formatting objects (XSL-FO) and the world's first output independent formatter. It is a *Java* application that reads a formatting object (FO) tree and renders the resulting pages to a specified output. Output formats currently supported include PDF, PCL, PS, SVG, XML, Print, AWT, MIF and Text. The default output target is PDF.

You can get *fop* from this link: http://xmlgraphics.apache.org/fop/ [http://xmlgraphics.apache.org/fop/]. Presently only *fop version 0.20.5* works with current DocBook tools.

Java

The *fop* processor is a *Java* application, so it needs a Java runtime to work. This is installed on many platforms by default, but unfortunately *fop* doesn't work with the GCJ [http://gcc.gnu.org/java/] runtime. There is apparently work being done on FOP to make it usable, but for now, this means installing Sun Java [http://java.sun.com/].

In addition to the default *j2re* package, you also need to install *JAI*, the Java Advanced Imaging library. You can get *JAI* from this link [http://java.sun.com/products/java-media/jai/iio.html]. *JAI* is not required, and the *PDF* file will be generated. It will just be missing all the graphics.

Fop also requires an environment variable to be set. This is JAVA_HOME. This needs to point to the top directory where your *Sun j2re* is installed. If you have any other problems with your Java installation, you can also try adding the *Sun j2re* path to the *CLASSPATH* environment variable.

Configuring The Code

Gnash uses GNU Autoconf (http://www.gnu.org/software/autoconf [http://www.gnu.org/software/autoconf/]) for configuration. All the standard configure options apply.

The *configure* script is not included in the CVS sources. It must be produced by running the *./autogen.sh* script in the top level source drirectory. This script requires you have *autoconf*, *automake*, and *libtool* installed. After building this script, you can configure Gnash by running it from the source directory like this: *./configure*, or from another directory by specifying the full path to configure.

By default, `make install' will install all the files in `/usr/local/bin', `/usr/local/lib' etc. You can specify an installation prefix other than `/usr/local' using `--prefix', for instance `--prefix=$HOME'.

Occasionally the default paths for a development package aren't correct. There are several options which can be used to adjust these paths to your own setup. By default, Gnash will look in all the standard places for these tools.

A configure script can support many options, but they fall into two main types. The first type are options to enable or disable features. The second type is used to specify custom paths for development packages which can't be found by the default searching and testing.

Specifying A Custom Path

This set of options typically use a *--with-[name]* naming convention. A Prefix can often be supplied, which is the top level directory which can be used to look for the other sub directories. Most options of this type have two variations, one to specify a path to the header files, and another to specify a path to

the libraries. This lets you override the default paths *configure* finds, or specify your own paths.

By default, none of the options should be required unless you want Gnash to use a specific version of a development package, or the configure test for Gnash fails to find the component. There are a lot of options, but Gnash has a lot of dependencies. If you find a configure test is failing on your machine, please submit a patch [https://savannah.gnu.org/bugs/?group=gnash] or file a bug report [https://savannah.gnu.org/bugs/?group=gnash].

--x-includes=DIR	--x-includes=DIR X include files are in DIR.
--x-libraries=DIR	--x-libraries=DIR X library files are in DIR.
--with-libxml=PFX	--with-libxml=PFX Prefix to where libxml is installed.
--with-libxml-libraries=DIR	--with-libxml-libraries=DIR Directory where libxml library is installed.
--with-libxml-includes=DIR	--with-libxml-includes=DIR Directory where libxml header files are installed.
--with-docbook=DIR	--with-docbook=DIR Directory where the DocBook style-sheets are installed.
--with-sdl-prefix=PFX	--with-sdl-prefix=PFX Prefix where SDL is installed.
--with-zlib-incl	--with-zlib-incl Directory where zlib header is installed.
--with-zlib-lib	--with-zlib-lib Directory where zlib library is installed.
--with-jpeg-incl	--with-jpeg-incl Directory where jpeg header is installed.
--with-jpeg-lib	--with-jpeg-lib Directory where jpeg library is installed.
--with-png-incl	--with-png-incl Directory where png header is installed.
--with-png-lib	--with-png-lib Directory where png library is installed.
--with-qt-dir	--with-qt-dir Directory where QT is installed. This is only used by the Klash plugin.
--with-qt-includes	--with-qt-includes Directory where the QT header files are installed. This is only used by the Klash plugin.
--with-qt-libraries	--with-qt-libraries

	Directory wherethe QT libraries are installed. This is only used by the Klash plugin.
--with-plugindir	--with-plugindir This is the directory to install the Firefox plugin in.
--with-ming	--with-ming Ming is used to build testcases, but not by the Gnash player itself.
--with-mad_incl	--with-mad_incl Directory where libmad header is installed.
--with-mad_lib	--with-mad_lib Directory where libmad library is installed.
--with-ogg_incl	--with-ogg_incl Directory where the libogg headers are installed.
--with-ogg_lib	--with-ogg_lib Directory where the libogg library is installed.
--with-gstreamer-incl	--with-gstreamer-incl Directory where the Gstreamer headers are installed. Gstreamer version 0.10 or greater must be used.
--with-gstreamer-lib	--with-gstreamer-lib Directory where the Gstreamer library is installed. Gstreamer version 0.10 or greater must be used.
--with-opengl-includes	--with-opengl-includes Directory where OpenGL (libMesa) headers are installed.
--with-opengl-lib	--with-opengl-lib Directory where the OpenGL (libMesa) library is installed.
--with-glext-incl	--with-glext-incl Directory where GtkGlExt headers are installed.
--with-glext-lib	--with-glext-lib Directory where the GtkGlExt library is installed.
--with-gtk2-incl	--with-gtk2-incl Directory where the Gtk2 headers are installed.
--with-gtk2-lib	--with-gtk2-lib Directory where the Gtk2 library is installed.

--with-cairo_incl	--with-cairo_incl Directory where the Cairo headers are installed.
--with-cairo-lib	--with-cairo-lib Directory where the Cairo library is installed.
--with-glib-incl	--with-glib-incl Directory where the Glib headers are installed.
--with-glib-lib	--with-glib-lib Directory where the Glib library is installed.
--with-pango-incl	--with-pango-incl Directory where the Pango headers are installed.
--with-pango-lib	--with-pango-lib Directory where the Pango library is installed.
--with-atk-incl	--with-atk-incl Directory where the ATK headers are installed.
--with-atk-lib	--with-atk-lib Directory where the ATK library is installed.
--with-pthread-incl	--with-pthread-incl Directory where the Pthread headers are installed.
--with-pthread-lib	--with-pthread-lib Directory where the Pthread library is installed.
--with-agg-incl	--with-agg-incl Directory where the AGG (Antigrain) headers are installed.
--with-agg-lib	--with-agg-lib Directory where the AGG (Antigrain) library is installed.
--with-ffmpeg-incl	--with-ffmpeg-incl Directory where the FFMPEG headers are installed.
--with-ffmpeg-lib	--with-ffmpeg-lib Directory where the FFMPEG library is installed.
--with-boost-incl	--with-boost-incl Directory where the Boost headers are installed.

--with-boost-lib	--with-boost-lib Directory where the Boost library is installed.
--with-curl-incl	--with-curl-incl Directory where the libCurl headers are installed.
--with-curl-lib	--with-curl-lib Directory where the libCurl library is installed.

Configure Options

In addition to being able to specify your the directories for various sub-components, there are also some switches which can be used at configuration time to enable or disable various features of Gnash. There are defaults for all of these options. These are typically used only by developers who don't have all the other development packages installed, and want to limit what is required for a quite build of *Gnash*.

--disable-xml	--disable-xml Disable support for XML. The default is to support XML.
--disable-fork	--disable-fork Disable the plugin forking the standalone player,and using a thread for the player instead. Currently forking the standalone player will give you the best results.
--enable-plugin	--enable-plugin Enable building the plugin. By default the Mozilla Firefox plugin won't be built, even if all the required files are found by configure. Configure *--with-plugindir=* to specify where the plugin should be installed.
--disable-glext	--disable-glext Disable using GtkGlExt, which forces the use of SDL instead. By default if the GtkGL extension for Gtk is found by configure, the GTK enabled GUI is built.
--disble-klash	--disble-klash Disble support for Konqueror plugin. If *--enable--plugin* is specified, and support for building KDE programs is found, Klash is built by default. This option limits the plugin to only the Mozilla/Firefox one.
--enable-libsuffix	--enable-libsuffix /lib directory suffix (64,32,none=default). This is only used by Klash.
--enable-embedded	--enable-embedded Link to Qt-embedded, don't use X. This is only used by Klash.

--enable-qtopia	--enable-qtopia Link to Qt-embedded, link to the Qtopia Environment. This is only used by Klash.
--enable-mac	--enable-mac Link to Qt/Mac (don't use X). This is only used by Klash.
--enable-sdk-install	--enable-sdk-install Enable installing the libraries and headers as an SDK.
--enable-testing	--enable-testing Enable testing-specific methods.
--enable-strict	--enable-strict Turn on tons of GCC compiler warnings. By default only *-Wall* is used with GCC.
--enable-ghelp	--enable-ghelp Enable support for the GNOME help system.
--enable-dom	--enable-dom When using the XML library, parse the messages using a DOM based parser. This is the default.
--enable-xmlreader	--enable-xmlreader When using the XML library, parse the messages using a SAX based parser.
--enable-dmalloc	--enable-dmalloc Enable support for the DMalloc memory debugging tool.
--enable-i810-lod-bias	--enable-i810-lod-bias Enable fix for Intel 810 LOD bias problem. Older versions of libMesa on the Intel i810 or i815 graphics processor need this flag or Gnash will core dump. This has been fixed in newer versions (summer 2005) of libMesa.
--enable-sound=gst|sdl	--enable-sound=gst|sdl Enable support for the a sound handing system. Currently only *sdl* works sufficiently. This is enabled by default when building Gnash.
--with-mp3-decoder=ffmpeg|mad	--with-mp3-decoder=ffmpeg|mad Specified the mp3 decoder to use with sdl sound handler. Mixing this with --enable-sound=gst is invalid. Using *mad* is the default decoder.
--enable-renderer=opengl|cairo|agg	--enable-renderer=opengl|cairo|agg Enable support for the a graphics backend. Currently only *opengl* and *agg* work suffi-

ciently. OpenGL is used when you have hardware accelerated graphics. AGG i used when you don't have hardware accelerated graphics. Tyically most desktop machines have OpenGL support, and most embedded systems don't. OpenGl is the default when building Gnash.

You can control other flags used for compiling using environment variables. Set these variables before configuring, and they will be used by the configure process instead of the default values.

CXX — CXX
C++ compiler command.

LDFLAGS — LDFLAGS
linker flags, e.g. -L[library directory] if you have libraries in a nonstandard directory.

CPPFLAGS — CPPFLAGS
C/C++ preprocessor flags, e.g. -I[headers directory] if you have headers in a nonstandard directory.

CXXFLAGS — CXXFLAGS
C++ compiler flags.

Cross Compiling And Configuration

To cross configure and compile Gnash, you first need to build a target system on your workstation. This includes cross compilers for the target architecture, and typically some system headers. You will also need *libxml2*, *libpng*, *libjpeg*, *sdl*, *opengl*, and *ogg* development packages built for the target system.

If you need to build up a target system from scratch, there is a good document and shell script at this web site: http://frank.harvard.edu/~coldwell/toolchain/ [http://frank.harvard.edu/~coldwell/toolchain/].

After I built up an ARM system in /usr/arm using the shell script from this web site, I then cross compiled all the other libraries I needed. The fun part is trying to get libMesa to cross compile, because it's not really set up for that.

So to build for an ARM based system on an x86 based systems, configure like this:

```
../gnash/configure --target=arm-unknown-linux-gnu --prefix=/usr/arm --host
```

The important options here are the ones that specify the architectures for the build.

--target — --target
The target architecture. This is the architecture the final executables are supposed to run on.

--host — --host
The host architecture. This is the architecture the executables are supposed to run on. This is usually the same as *--target* except when building a compiler as a Canadian Cross. This is when you build a cross compiler on a Unix machine, that runs on

a *win32* machine, producing code for yet a third architecture, like the *ARM*.

--build --build
This is the system this build is running on.

Building Gnash

After managing to configure Gnash, building the code is simple. Gnash is built using *GNU make*.

Compiling The Code

After configuring, typing *make* will compile the code. No options are necessary. If desired, you can re-define the variables used by *make* on the command line when invoking the program. The few flags of interest are *CFLAGS* and *CXXFLAGS*, often used to turn on debugging or turn off optimizing. Invoking make as in this example would build all the code with debugging turned on, and optimizing turned off. The default values for both of these variables is *-O2 -g*.

```
make CFLAGS=-g CXXFLAGS=-g
```

If the compilation ends with an error, check the output of configure and make sure nothing required to build Gnash is missing.

Processing The Documentation

By default, the documentation isn't built at all. It isn't even built when typing *make install* from the top level build directory. It's only built when specified with a specific target in the generated *Makefile* in the *doc/C/* sub-directory. All the docs are built in this directory when executing a *make install*.

There is a target for each output format, *make html*, *make pdf*, *make info*, and *make man*. A higher level target, *make alldocs*, builds the four main formats for the documentation.

Gnash also has support to use Doxygen [http://www.stack.nl/~dimitri/doxygen/index.html] to produce *HTML* pages documenting the internals of Gnash. While this is not necessarily internals documentation, it does give very useful information about all the files, the classes, a cross reference, and other data.

You need to have Doxygen installed to produce these docs. If you do have it installed, typing *make apidoc* in the *doc* directory will make these docs under a sub directory of *apidoc/html*

Installation

Gnash installs its libraries so they can be found in the runtime path for the Gnash executable. Unless the --prefix option is used at configuration time, the libraries get installed in /usr/local/lib. If you install Gnash in a non-standard location, you have to specify this runtime path by one of two means.

The traditional way that works on all Unix platforms is to set the LD_LIBRARY_PATH environment variable to $prefix/lib. You can have multiple paths in this variable as long as they are separated by a colon ":" character.

For GNU/Linux systems, the custom path to the libraries can be added to the /etc/ld.so.conf file. After adding the custom path, then run (as root) the *ldconfig* command to update the runtime cache.

What Code Gets Installed and Where

Several libraries get installed, as well as the three executables. All the libraries, *libbase, libgeometry, libgbackend, libserver, and libmozsdk* get installed in the directory pointed to by *$prefix*. This variable is set by the *--prefix* option at configure time, and if not specified, it defaults to */usr/local*. All the libraries get installed in *$prefix/lib* where most packages also install their libraries.

The plugin gets installed in the plugins directory of the version of the*Firefox* or *Mozilla* you have the development packaged installed for. For builds from Mozilla CVS, the default installation directory is */usr/local/lib/firefox-[version number]/plugins/*. The default system directory used when installing packages is */usr/lib/mozilla/plugins*. Note that you have to be *root* to install files in a system directory. For some reason when the plugin is installed in the users *$HOME/.mozilla/plugins* or *$HOME/.firefox/plugins* directory, unresolved symbols from deep within Firefox appear.

The executables get installed in a *bin* directory of the directory specified by *$prefix*. Once again, this path defaults to */usr/local/bin* if a special prefix wasn't configured in.

If using a single file-system *NFS*mounted to multiple platforms, you can specify an additional option, *--exec-prefix*. This is where all the platform dependent executables and libraries can get installed.

What Documentation Gets Installed and Where

The documentation only installs when GNOME Help support is enabled by using *--enable-ghelp*. Because GNOME help files get installed in a system directory when building from source, you need to either change the permissions on the destination directory, or do the install as *root*. The default directory for GNOME Help files is: */usr/local/share/gnash/doc/gnash/C/*.

A config file in the Gnash source tree, *doc/C/gnash.omf* is used to specify under which menu item Gnash is listed in the GNOME Help system.

Software Internals

A Tour of Gnash

The top level of Gnash has several libraries, *libgnashbase*, *libgnashgeo*, *libgnashserver*, *libgasnhasobjs* and *libgnashbackend*. There are two utility programs included for debug parsing and processing of Flash movie files which test the Actionscript interpreter. There is also a standalone flash movie player.

The Libraries

libgnashbase

Libgnashbase contains support classes used by the rest of the code. Among these classes is a small and efficient STL library clone that uses smart pointers. A current goal is to replace this small STL clone with standard STL containers to reduce the amount of code which has to be maintained, and to add functionality not currently in the smaller implementation.

Gnash makes heavy use of smart pointers, so memory allocations are freed up automatically by the interpreter.

libgnashgeo

Libgnashgeo contains code for device independent graphics routines.

libgnashui

Libgnashgui contains code for a portable GUI class that suppots using GTK2, a framebuffer, SDL, or

KDE.

libgnashserver

Libgnashserver is the guts of the interpreter itself. This is where the main code for the interpreter lives.

libgnashasobjs

Libgnashasobjs contains all the ActionScript classes used by the interpreter.

libgnashamf

AMF is the data format used internally by Flash. This is Gnash's support library to handle AMF data. This is currently unused, but when the LocalConnection class is more fully implemented, this will be used to transfer data between flash movies.

libgnashbackend

Libgnashbackend is a small library containing OpenGL and SDL code that glues this display to the Gnash display.

libgnashplugin

Libgnashplugin is the Mozilla/Firefox plugin.

libklashpart

Libklashpart is the Konqueror plugin.

The Applications

There are currently a few standalone programs in Gnash, which serve to either assist with Gnash development or play flash movies.

The Standalone Player

This is the standalone OpenGL back-end used to play movies. There are several command-line options and keyboard control keys used by Gnash which are documented here.

Gparser

Gparser uses the Gnash parser to disassemble the flash movie, and dumps the object types, the depth, and other information to make sure Gnash is parsing the file correctly.

Gprocesser

Gprocesser is used to print out the actions (using the -va option) or the parsing (using the -vp option) of a flash movie. It is also used to produce the *.gsc* files that Gnash uses to cache data, thereby speeding up the loading of files.

The Plugin

The plugin is designed to work within Mozilla or Firefox, although there is Konqueror support as well. The plugin uses the Mozilla NSPR plugin API to be cross platform, and is portable, as well as being well integrated into Mozilla based browsers.

One future thought for the plugin is to use the new Firefox 1.5 or greater version of Firefox. This version

has added a drawable canvas window that supports hardware acceleration, and is designed to support things like rendering directly into the canvas without needing OpenGL.

Current Status

As of March 30, 2006, the plugin works! This works in a fashion similar to MozPlugger in that the standalone player is used instead of using a thread. This gets around the issue of having to maintain a seperate player to support the plugin. It also gets around the other issues that Gnash itself is not thread safe at this time.

There are a few limitations in the current implementation, but it works well enough to be used for web surfing. The main limitations are the SDL version has no event handling, and sound doesn't work yet.

GUI Support

Any plugin that wants to display in a browser window needs to be tied into the windowing system of the platform being used. On GNU/Linux systems, Firefox is a GTK2+ application. There is also KDE support through the use of the Klash plugin.

Gnash can use either SDL or GTK now to create the window, and to handle events for the standalone player. Work is underway to add a portable interface for more windowing toolkits to allow better embedded device support when running in framebuffer only devices.

The SDL version is more limited, but runs on all platforms, including win32. It has no support for event handling, which means mouse clicks, keyboard presses, and window resizing doesn't work. I personally find the default event handler slow and unresponsive. Gnash has support to use fast events, (currently not enabled) which is an SDL hack using a background thread to pump events into the SDL event queue at a much higher rate.

There are a variety of development libraries that build a GUI widget system on top of SDL and OpenGL. The use of these to add menus and dialog boxes to the SDL version is being considered.

The GTK support is currently the most functional, and the best integrated into Firefox. The performance of this version is better than the SDL version because of the more efficient event handling within GTK. For the best end user experience, use the GTK enabled version.

GTK also allows Gnash to have menus and dialog boxes. Currently this is only being utilized in a limited fashion for now. There is a right mouse button menu that allows the user to control the movie being player the same way the existing keyboard commands do.

Mozplugger

Mozplugger [http://mozplugger.mozdev.org/]. is a *Mozilla/Firefox* plugin that uses external programs to play video, audio, and other multimedia content in the browser. With some support added to the external application, it's possible to force the external program to use the internal window in the browser where this plugin is supposed to display. This enables one to then run the standalone player and display its output in the browser.

While this is not an optimal solution, it does enable one to use Gnash as the flash player when browsing. The main issue appears to be that the Flash movie being played doesn't get any mouse or keyboard input. That may be a mozplugger configuration issue, however.

Use of MozPlugger is obsolete now that the Gnash plugin works. Still, this may be useful still on some platforms.

Add this to your *$(HOME)/.mozilla/mozpluggerrc* file to enable this:

```
    application/x-shockwave-flash:swf:Shockwave Gnash
nokill embed noisy ignore_errors hidden fill swallow(Gnash) loop: gnash -v
: gnash -v "$file" -x $window
```

Once this is added, you must delete the *$(HOME)/.mozilla/firefox/pluginreg.dat* file to force Firefox to register the plugins again. This is an ascii text file, so if the patch has been added correctly, you'll see an entry for *swf* files after it is recreated. You will need to restart Firefox to recreate this file.

This file is not recreated immediately when restarting Firefox, but waits till the first time a plugin is used. You can force creation of this file by typing *about:plugins* into the URL entry of the browser window. The output will also contain information about the mozplugger. You should see an entry for Gnash now.

Klash

Klash is MozPlugger type support for KDE's Konqueror web browser. Klash makes Gnash a *kpart*, so it's integrated into KDE better than when using MozPlugger. Klash uses the standalone player, utilizing Gnash's "-x" window plugin command line option.

By default, Klash is not built. To enable building Klash, use the *--enable-klash* option when configuring. Other than installing, there is nothing else that needs to be done to install Klash.

Mozilla/Firefox Plugin

The Mozilla SDK has two API layers for plugins. The older layer is documented in the Geeko Plugin API Reference [http://www.gnu.org/software/gnash/manual/plugin.pdf], and the newer layer doesn't appear to be documented. The new API is simpler, and is portable across multiple versions of Mozilla or Firefox. The new API is just a layer on top of the older one, so this manual still applies.

Most of the programming of a plugin is filling in real emphasis for the standard API functions and methods. Firefox uses these to create the plugin, and to send it data.

When initializing or destroying a plugin, no matter how many instances are being used, the C API is used. These functions are typically called once for each plugin that is loaded.

Plugin C API

The lower layer is a C based API which is used by Firefox to initialize and destroy a plugin. This is so a plugin can be portable across multiple systems, since C++ emphasis is not portable between most C++ compilers. This where most of the behind the scenes work is done in a plugin. For Gnash, the sources this lower layer are in *plugin/mozilla-sdk*. They were added to the Gnash source tree so it wouldn't be necessary to have the Mozilla development packages installed to compile the Gnash plugin.

This is also the older API used for plugins, so is usually the one used if you dig around for plugin examples on the web. These are the main functions which have to be implemented in a plugin for it to be recognized by the browser, and to be initialized and destroyed.

NS_PluginInitialize	NS_PluginInitialize This C function gets called once when the plugin is loaded, regardless of how many instantiations there are actually playing movies. So this is where all the one time only initialization stuff goes that is shared by all the threads.
NS_NewPluginInstance	NS_NewPluginInstance

	This instantiates a new object for the browser. Returning a pointer to the C++ plugin object is what ties the C++ and C emphasis parts of the API together.
NS_DestroyPluginInstance	NS_DestroyPluginInstance This destroys our instantiated object when the browser is done.
NS_PluginShutdown	NS_PluginShutdown This is called when a plugin is shut down, so this is where all the one time only shutdown stuff goes.
NPP_GetMIMEDescription	NPP_GetMIMEDescription This is called to get the MIME types the plugin supports.
NS_PluginGetValue	NS_PluginGetValue This is used by Firefox to query information from the plugin, like the supported MIME type, the version number, and a description.

Plugin C++ API

The higher level layer is the one we are most concerned with. This is an instantiation of the *nsPluginInstanceBase* class, as defined by the Mozilla SDK, for our plugin. With this API, a plugin is mostly defining the standard entry points for Firefox, and the emphasis that implements the glue between the Firefox and our plugin.

These are called for each instantiation of plugin. If there are three Flash movies on a web page, then three instances are created. Unfortunately for plugin programmers, these functions may randomly be called more than once, so it's good to use initialization flags for things that should only be done one per thread. For instance, *nsPluginInstance::init()* and *nsPluginInstance::SetWindow()* are called more than once, so the plugin must protect against actions that could be destructive.

nsPluginInstance::nsPluginInstance	nsPluginInstance::nsPluginInstance Create a new plugin object.
nsPluginInstance::init	nsPluginInstance::init This methods initializes the plugin object, and is called for every movie which gets played. This is where the thread-specific information goes.
nsPluginInstance::SetWindow	nsPluginInstance::SetWindow This sets up the window the plugin is supposed to render into. This calls passes in various information used by the plugin to setup the window. This may get called multiple times by each instantiated object, so it can't do much but window specific setup here. This is where the main emphasis is that sets up the window for the plugin.
nsPluginInstance::NewStream	nsPluginInstance::NewStream Opens a new incoming data stream, which is

	the flash movie we want to play. A URL can be pretty ugly, like in this example: http://www.sickwave.com/swf/navbar/navbar_sw.swf?atfilms=http%3a//www.atm.com/af/home/ &shickwave=http%3a//www.sickwave.com &gblst=http%3a//gbst.sickwave.com/gb/gbHome.jsp&known=0 ../flash/gui.swf?ip_addr=foobar.com&ip_port=3660&show_cursor=true&path_prefix=../flash/&trapallkeys=true" So this is where we parse the URL to get all the options passed in when invoking the plugin.
nsPluginInstance::Write	nsPluginInstance::Write Read the data stream from Mozilla/Firefox. For now we read the bytes and write them to a disk file.
nsPluginInstance::WriteReady	nsPluginInstance::WriteReady Return how many bytes we can read into the buffer.
nsPluginInstance::DestroyStream	nsPluginInstance::DestroyStream Destroy the data stream we've been reading. For Gnash, when the stream is destroyed means we've grabbed the entire movie. So we signal the thread to start reading and playing the movie.
nsPluginInstance::shut	nsPluginInstance::shut This is where the movie playing specific shutdown emphasis goes.
nsPluginInstance::~nsPluginInstance	nsPluginInstance::~nsPluginInstance This destroys our plugin object.
NS_PluginInitialize::initGL	NS_PluginInitialize::initGL This is a Gnash internal function that sets up OpenGL.
NS_PluginInitialize::destroyContext	NS_PluginInitialize::destroyContext This is a Gnash internal function that destroys a GLX context.
nsPluginInstance::getVersion	nsPluginInstance::getVersion This returns the version of Mozilla this plugin supports.
nsPluginInstance::GetValue	nsPluginInstance::GetValue This returns information to the browser about the plugin's name and description.
nsPluginInstance::URLNotify	nsPluginInstance::URLNotify

OpenGL and Threads

Neither OpenGL nor X11 has any built-in support for threads. Most actions aren't even atomic, so care

has to be made to not corrupt any internal data. While it is difficult to render OpenGL from multiple threads, it can be done with the proper locking. The downside is the locking adds a performance hit, since all the threads will have to have the access synchronized by using mutexes.

The X11 context is maintained one per instantiation of the plugin. It is necessary to lock access to the X11 context when using threads by using *XLockDisplay()* and *XUnlockDisplay()*. A connection to the X11 server is opened for every instantiation of the plugin using *XOpenDisplay()*.

The *The GLX Context* is maintained one per instantiation of the plugin for a web page. If there are more than one Flash movie, there is more than one GLX Context. A GLX context can be created by using *glXCreateContext()*, and then later destroyed by using *glXDestroyContext()*. When swapping threads, the context is changed using *glXMakeCurrent()*.

All the emphasis that directly accesses a GLX context or the X11 display must be wrapped with a mutex.

Plugin Event Handling

Firefox on most unix systems is a GTK+ application, so it is possible to have the plugin hook into the X11 event handling via GLX or GTK. Since Firefox uses GTK, so does Gnash. This also allows the addition of a right-click mouse menu for controlling the player. The GTK build of Gnash offers the best browsing experience as it's more functional than the SDL version.

It is also possible to disable the *GTK* support so only the older *SDL* support is used. In this case Gnash can't support event handling within the browser. This means that when using the SDL of the plugin, mouse clicks and keys pressed get ignored. Windows also can't be resized, and sometimes they overrun their boundaries as well. To disable the GTK support and force SDL to be used anyway, configure with *--disable-glext*

The Debug Logging System

Gnash supports a debug logging system which supports both C and C++ natively. This means you can use both *printf()* style debug messages and C++ *iostreams* style, where you can print C++ objects directly as you would when using *cout*.

In the beginning, Gnash only supported the C API for debug logging, so it is the most heavily used in Gnash. This API was used in the *log_msg()* and *log_error()* functions, and used a callback to set them up.

It became apparent one day the callback was never needed, and I got tired of having to use *c_str()* on string data just to print them out.

If a filename is not specified at object construction time, a default name of *gnash-dbg.log* is used. If Gnash is started from the command line, the debug log will be created in the current directory. When executing Gnash from a launcher under *GNOME* or *KDE* the debug file goes in your home directory, since that's considered the current directory.

There is common functionality between using the C or C++ API. Optional output is based on flags that can be set or unset. Multiple levels of verbosity are supported, so you can get more output by supplying multiple *-v* options on the command line. You can also disable the creation of the debug log.

Logging System C API

These functions are clones of the originals as they were used for Gnash. These function the same as always except outout can be logged to disk now as well. These currently print no timestamp with the output, which is the older functionality. As these functions are implemented on top of the C++ API now, they can be used without corrupting the output buffers.

void log_msg(const char* fmt, ...)	void log_msg(const char* fmt, ...) Display a message if verbose output is enabled. By default the messages are always written to the disk file, but optionally displayed in the terminal.
log_error(const char* fmt, ...)	log_error(const char* fmt, ...) Display an error message if verbose output is enabled. By default the error messages are always written to the disk file, but optionally displayed in the terminal.
log_warning(const char* fmt, ...)	log_warning(const char* fmt, ...) Display a warning message if verbose output is enabled. By default the error messages are always written to the disk file, but optionally displayed in the terminal.

Logging System C++ API

This is the new C++ streams based API that can be used to print C++ objects natively. All output lines are timestamped.

There are two macros used for program tracing. these can be used in both C or C++ code with one little difference. Since C doesn't have destructors, you must call *GNASH_REPORT_RETURN* at the end of a function to display the function returning message.

GNASH_REPORT_FUNCTION;	GNASH_REPORT_FUNCTION; When this is included in a C++ method, a message is printed when entering and exiting this method by hooking into the constructor and destructor. These are always written to the disk file, but optionally written to the screen only at the highest levels of verbosity.
GNASH_REPORT_RETURN;	GNASH_REPORT_RETURN; This is used by C functions to print the returning from function debug message. For C++, this macro is executed automatically by the destructor.

This is the main API for the logging system. By default everything is setup to write to the default *gnashdbg.log* file whenever a verbose option is supplied. Optionally it is possible to open a log file with a specified name, allowing multiple output files.

openLog(const char *filespec)	openLog(const char *filespec) Open the debug file with the name specified by *filespec*. This file goes in the current directory, or your home directory if using a menu based launcher.
closeLog(void)	closeLog(void) Close a debug log. The disk file remains.

removeLog(void)	removeLog(void) Delete the debug log file from disk.
setVerbosity(void)	setVerbosity(void) Increment the verbosity level.
setVerbosity(int)	setVerbosity(int) Set the verbosity level.
setStamp(bool flag)	setStamp(bool flag) If *flag* is *true*, then print a timestamp prefixed to every output line. If *flag* is *false*, then don't print a timestamp.
setWriteDisk(bool flag)	setWriteDisk(bool flag) If *flag* is *true*, then create the disk file. If *flag* is *false*, then don't create the disk file.

ActionScript Support

Adding New ActionScript Classes

Adding a new ActionScript class is a relatively simple process. A new file is created to hold the code, with an associated header file. The file name is usually the name of the ActionScript class itself, something like *XML*. All implementations are written in *C++*. In the CVS source tree for Gnash, there is a utility file called *gen-files.sh* that can be used to generate a template for a new ActionScript class. At this time templates have been generated for all documented ActionScript classes.

Defining a new ActionScript Class

The header file defines the class and its methods. The symbol name used to look up the code which implements the ActionScript class is added later.

Each class needs an associated version which is a derived form of the *as_object* class used to internally represent objects in Gnash. At its simplest, this structure just encapsulates an object of the desired class.

```
class Foo {
    public:
        foo() {};
        ~foo() {};
        bool GetBar() { return _bar; }
    private:
        bool _bar;
}
struct foo_as_object : public gnash::as_object {
    Foo obj;
}
```

The *obj* is an instantiation of the data for this object. It isn't required unless this object needs internal data that has to stay resident in the player.

Whenever this object is being handed the code for this class, it is initially accessed by its derived binding. Internal data for this object can be accessed through the *obj*.

```
foo_as_object *foo = env.top(0).to_object();
```

```
bool result = foo->obj.GetBar();
```

A more complex example might be to add hooks to the constructor an destructor for the class to keep track of memory allocations and cleanup. In this case only a debug statement is printed.

```
struct foo_as_object : public gnash::as_object {
    Foo obj;
    foo_as_object() {
        log_msg("\tCreating foo_as_object at %p \n", this);
    };
    ~foo_as_object() {
        log_msg("\tDeleting foo_as_object at %p \n", this);
    };
};
```

An even more complex example might be to add hooks to how the list of member of a class is kept. The element *m_members*, is a list of each symbol name and its associated code. Normally this is kept internally within the interpreter engine, but in this example for certain methods we want to return a point to itself, instead of getting the data from the list.

```
struct xml_as_object : public gnash::as_object {
    XML obj;
    xmlnode_as_object() {
        log_msg("\tCreating xmlnode_as_object at %p \n", this);
    };
    ~xmlnode_as_object() {
        log_msg("\tDeleting xmlnode_as_object at %p \n", this);
    };
    virtual bool get_member(const tu_stringi& name, as_value* val) {
        if ((name == "firstChild") || (name == "childNodes")) {
            val->set_as_object_interface(this);
            return true;
        }
        if (m_members.get(name, val) == false) {
            if (m_prototype != NULL) {
                return m_prototype->get_member(name, val);
            }
            return false;
        }
      return true;
   }
};
```

Instantiating a new Class

To add a new object to the list maintained by Gnash, it needs to be added to the function *gnash::action_init()*, in *action.cpp*. The symbol name is specified along with a function pointer that will be called when the symbol name is seen by the interpreter.

```
obj->set_member("XML", as_value(xml_new));
```

The function used to instantiate a new object is passed the creation data in a *fn_call* data structure. This is used to pass data both into and returned from this function.

The *fn_call* data structure has several methods for operating on the data for the function. *fn_call::nargs* is a variable that soecifies how many arguments are being passed in. Al the arguments are on a stack. To pop an argument off the stack, use *fn.env->top(0)*. In this case popping the first argument off the stack.

The object popped off the stack also has its own methods. The main one of interest is *get_type*. This returns the type of the object being referred to.

```
if (fn.env->top(0).get_type() == as_value::STRING) {
   ...
}
```

The supported data types for an object are *BOOLEAN*, *STRING*, *NUMBER*, *OBJECT*, *C_FUNCTION*, *AS_FUNCTION*. Because they are defined as part of the as_value class, they need to always have the class name prefixed to use these as a constant. You can retrieve the value of an *as_value* using the conversion methods. For example, *to_tu_string* returns the value as string using the Gnash small STL library. Similarly, *to_number* would return this same value as a *double*.

To add methods to the class, a new class needs to be instantiated as an object. Each ActionScript object can have child methods attached in a similar way as the object was. In this case, the built-in *set_member* function is used.

```
xml_obj = new xml_as_object;
xml_obj->set_member("load", &xml_load);
```

To make the object active within the interpreter, the new object gets returned by the function using the *fn_call* typed parameter.

```
fn.result->set_as_object_interface(xml_obj);
```

A complete example of a function used to instantiate a new ActionScript object is as follows. This example also calls internally defined methods in the class, in this case to process and XML file, and to create the parsed XML tree.

```
void
xml_new(const fn_call& fn) {
    as_value      inum;
    xml_as_object *xml_obj;

    if (fn.nargs > 0) {
        if (fn.env->top(0).get_type() == as_value::STRING) {
            xml_obj = new xml_as_object;
            tu_string datain = fn.env->top(0).to_tu_string();
            xml_obj->obj.parseXML(datain);
            xml_obj->obj.setupFrame(xml_obj, xml_obj->obj.firstChild()
        } else {
            xml_as_object *xml_obj = (xml_as_object*)fn.env->top(0).to
            fn.result->set_as_object_interface(xml_obj);
            return;
```

```
        }
    } else {
        xml_obj = new xml_as_object;
        xml_obj->set_member("load", &xml_load);
        xml_obj->set_member("loaded", &xml_loaded);
    }
    fn.result->set_as_object_interface(xml_obj);
}
```

Adding a Property

Adding a new new property to an object is similar to adding a callback for a method. Instead of using a C function, a string or number is used.

```
as_obj->set_member("nodeName", as_value("HelloWorld"));
```

When a Flash movie looks this up as a property, the value can be found directly without a function call. This scrap of ActionScript code as compiled by Ming's *makeswf* compiler shows the difference.

```
// Call the hasChildNodes() function
if (node.hasChildNodes() == true) {
    trace("CHILDREN");
}
// Get the value of the nodeName property
if (node.nodeName == "HelloWorld") {
    trace("MATCHED");
}
```

Parameter Passing

Parameters are passed to the callback functions for a class's methods and properties using the *fn_call* data structure. This data structure contains all the incoming parameters for a callback, as well as it contains the final result from the callback to be passed back into the player.

Getting Parameter Data

Parameter data is passed on the stack in a similar way to any function call in any language. There are several fields of the *fn_call* data structure that get used in this example:

```
xml_as_object *xml_obj = (xml_as_object*)fn.this_ptr;
if (fn.nargs) {
    filespec = fn.env->bottom(fn.first_arg_bottom_index).to_string;
}
```

Using the *fn* variable which was passed as the sole parameter of the callback, we can access the data. The *fn.this_ptr* returns a reference to the class which is invoking this method. This is how the object data can be retrieved. The *fn.nargs* is a count of how many objects are being passed into the callback.

The current stack of the player is passed in via the *fn.env* field. This is used to access the data passes to this callback. To find the location of this particular stack frame, the *fn.first_arg_bottom_index* field is used to point to the stack frame. More detail on the environment stack can be found here..

For this example, the stack has a *as_environment::top()* and a *as_environment::bottom()* that are used to pull arguments off the stack. When using *fn.first_arg_bottom_index*, the *as_environment::bottom()* method should be used as in the example.

The top of the stack for this frame can also be accessed using the *as_environment::top()* method. Top takes an integer index as to which value to retrieve,

```
if (fn.nargs > 0) {
 name = fn.env->top(0).to_string());
}
```

If the type of the object is needed, that can be accessed by using the *as_value::get_type()* method. There are more details on the types of values in the Handling Values section of this manual.

```
if (fn.nargs > 0) {
    if (fn.env->top(0).get_type() == as_value::STRING) {
        name = fn.env->top(0).to_string);
    }
    if (fn.env->top(0).get_type() == as_value::NUMBER) {
        value = fn.env->top(0).to_number);
    }
}
```

The internal data for this object can be accessed through the base class. Any data set using this object will stay resident in the player.

```
foo_as_object *foo_obj = (foo_as_object*)fn.this_ptr;
bool bar = foo_obj->obj.GetBar();
```

Returning Data

Data is also returned in the data structure passed to the callback. This example calls a method of the object passed in on the stack, and then sets the return code to be the return code of calling the method.

```
// Set the argument to the function event handler based on
// whether the load was successful or failed.
ret = xml_obj->obj.load(filespec);
fn.result->set_bool(ret);
```

The *result* field of the *fn* variable is a *gnash::as_value* object, so it's possible to set the value of several data types.

Here is a short list of the most often used data types returned from callbacks:

as_value::set_bool()	as_value::set_bool() Set the result to a boolean value.
as_value::set_int()	as_value::set_int()

	Set the result to an integer value.
as_value::set_double()	as_value::set_double() Set the result to a floating point double value.
as_value::set_string()	as_value::set_string() Set the result to a *const char** value.
as_value::set_as_object_interface()	as_value::set_as_object_interface() Set the result to an object value.

Level of Conformance

Gnash is a capable of reading SWF v7 files and opcodes. Not all ActionScript classes are implemented yet, but all of the most heavily used ones are. Many ActionScript classes are partially implemented; there is support for all of the commonly used methods of each class.

This is a quick list of what is currently implemented in Gnash as of January, 2006.

Class Name	**When Added**	**Conformance**
_global		Partially implemented.
Accessibility	swf v7 ???	Unimplemented.
Array	swf v6	Partially Implemented.
Boolean	swf v5	Fully implemented.
Button	swf v6	Most all the functionality is implemented.
Camera	swf v6	Unimplemented.
Color	swf v5	Unknown status.
ContextMenu	swf v7	Unimplemented.
CustomActions	swf v6	Unimplemented.
Date	swf v5	Partially implemented.
Error	swf v7	Unimplemented.
Function	swf v6	Unimplemented.
Key	swf v6	Fully implemented.

Class Name	When Added	Conformance
LoadVars	swf v6	Unimplemented.
LocalConnection	swf v6	Partially implemented.
Math	swf v4	Fully implemented.
Microphone	swf v6	Unimplemented.
Mouse	swf v5	Most all the functionality is implemented.
MovieClip	swf v3	Partially implemented.
MovieClipLoader	swf v7	Barely implemented.
NetConnection	swf v7	Partially implemented.
NetStream	swf v7	Unimplemented.
Number	swf v5	Partially implemented.
Object	swf v5	Partially implemented.
Selection	swf v5	Unimplemented.
SharedObject	swf v6	Unimplemented.
Sound	swf v5	Partially implemented.
Stage	swf v6	Unimplemented.
String	swf v5	Partially implemented.
System	swf v6	Unimplemented.
TextField	swf v6	Partially implemented.
TextFormat	swf v6	Partially implemented.
Video	swf v6	Unimplemented.

Class Name	When Added	Conformance
XML	swf v5	Almost fully implemented.
XMLNode	swf v5	Almost fully implemented.
XMLSocket	swf v5	Almost fully implemented.

Accessibility ActionScript Class

This class implements an Accessibility object.

The Methods of the Class

isActive()
isActive()
Return whether a screen reader is in use.

updateProperties()
updateProperties()
Change the display to use a screen reader.

Accessibility Class Conformance

Class Name	Conformance.
isActive()	This method is unimplemented.
updateProperties()	This method is unimplemented.

Array ActionScript Class

This class implements an array container.

The Methods of the Array Class

concat()
concat()
Concatenates two objects and returns a new array.

join()
join()
Join all the array elements into a string.

pop()
pop()
Return the last element in the array by removing if from the array.

push() push()
Push an element onto the end of the array.

reverse() reverse()
Reverse the order of the elements in an array.

shift() shift()
Return the first element in the array by removing if from the array.

sort() sort()
Sort the elements in the array.

sortOn() sortOn()
Sort the elements in the array based on one field.

Splice() Splice()
Return a subset of the array elements as a new array.

toString() toString()
Return the elements in an array as a string.

unshift() unshift()
Place an element on the front of the array.

The Properties of the Array Class

length length
The number of objects in the array.

Array Class Conformance

Class Name	**Conformance**
concat()	This method is implemented.
join()	This method is implemented.
pop()	This method is implemented.
push()	This method is implemented.
reverse()	This method is implemented.
shift()	This method is implemented.
slice()	This method is implemented.
sort()	This method is implemented.
sortOn()	This method is implemented.

Class Name	Conformance
splice()	This method is implemented.
toString()	This method is implemented.
unshift()	This method is implemented.
length	This property is implemented.

Boolean ActionScript Class

This class implements a Boolean object, the primary purpose of which is to return a string representation of a Boolean value.

The Methods of the Boolean Class

toString() toString()
Concatenates two objects and returns a new Boolean.

valueOf() valueOf()
Join all the Boolean elements into a string.

Boolean Class Conformance

Class Name	Conformance
toString()	This method is implemented.
valueOf()	This method is implemented.

Button ActionScript Class

This class implements an Button object.

The Methods of the Button Class

getDepth() getDepth()

The Properties of the Button Class

_alpha _alpha

enabled	enabled
_focusrect	_focusrect
_height	_height
menu	menu
_name	_name
_parent	_parent
_quality	_quality
_rotation	_rotation
_soundbuftime	_soundbuftime
tabEnabled	tabEnabled
tabIndex	tabIndex
_target	_target
trackAsMenu	trackAsMenu
_url	_url
useHandCursor	useHandCursor
_visible	_visible
_width	_width
_x	_x
_xmouse	_xmouse
_xscale	_xscale
_y	_y
_ymouse	_ymouse
_yscale	_yscale

The Event Handlers of the Button Class

onDragOut	onDragOut
onDragOver	onDragOver
onKeyDown	onKeyDown
onKeyUp	onKeyUp
onKillFocus	onKillFocus

onPress	onPress
onRelease	onRelease
onReleaseOutside	onReleaseOutside
onRollOut	onRollOut
onRollOver	onRollOver
onSetFocus	onSetFocus

Button Class Conformance

Class Name	Conformance
getDepth()	This method is implemented.
_alpha	This property is implemented.
enabled	This property has an unknown status.
_focusrect	This property is implemented.
_height	This property is implemented.
_menu	This property is unimplemented.
_name	This property is implemented.
_parent	This property is implemented.
_quality	This property is implemented.
_rotation	This property is implemented.
_soundbuftime	This property is implemented.
_tabEnabled	This property is unimplemented.
_tabIndex	This property is unimplemented.
_target	This property is implemented.
_trackAsMenu	This property is unimplemented.
_url	This property is implemented.
_useHandCursor	This property is unimplemented.

Class Name	Conformance
_visible	This property is implemented.
_width	This property is implemented.
_x	This property is implemented.
_xmouse	This property is implemented.
_xscale	This property is implemented.
_y	This property is implemented.
_ymouse	This property is implemented.
_yscale	This property is implemented.
onDragOut	This event handler is implemented.
onDragOver	This event handler is implemented.
onKeyDown	This event handler is implemented.
onKeyUp	This event handler is implemented.
onKillFocus	This event handler is not implemented.
onPress	This event handler is implemented.
onRelease	This event handler is implemented.
onReleaseOutside	This event handler is not implemented.
onRollOut	This event handler is not implemented.
onRollOver	This event handler is not implemented.
onSetDocus	This event handler is not implemented.

Camera ActionScript Class

This class implements an Camera object.

The Methods of the Class

get() get()

setMode()	setMode()
setMotionLevel()	setMotionLevel()
setQuality()	setQuality()

The Properties of the Camera Class

activityLevel	activityLevel
bandwidth	bandwidth
currentFps	currentFps
fps	fps
height	height
index	index
motionLevel	motionLevel
motionTimeOut	motionTimeOut
muted	muted
name	name
names	names
onActivity	onActivity
onStatus	onStatus
quality	quality
width	width

Camera Class Conformance

Class Name	**Conformance**
get()	This method is unimplemented.
setMode()	This method is unimplemented.
setMotionLevel()	This method is unimplemented.
setQuality()	This method is unimplemented.
activityLevel	This property is unimplemented.

Class Name	Conformance
bandwidth	This property is unimplemented.
currentFps	This property is unimplemented.
fps	This property is unimplemented.
height	This property is unimplemented.
index	This property is unimplemented.
motionLevel	This property is unimplemented.
motionTimeOut	This property is unimplemented.
muted	This property is unimplemented.
name	This property is unimplemented.
names	This property is unimplemented.
onActivity	This property is unimplemented.
onStatus	This property is unimplemented.
quality	This property is unimplemented.
width	This property is unimplemented.

Color ActionScript Class

This class implements an Color object.

The Methods of the Class

getRGB()	getRGB()
getTransform()	getTransform()
setRGB()	setRGB()
setTransform()	setTransform()

The Properties of the Color Class

Color Class Conformance

Class Name	Conformance
getRGB()	This method has an unknown status.
getTransform()	This method has an unknown status.
setRGB()	This method has an unknown status.
setTransform()	This method has an unknown status.

ContextMenu ActionScript Class

This class implements an ContextMenu object.

The Methods of the Class

copy()
copy()
Return a copy of the menu.

hideBuiltInItems()
hideBuiltInItems()
Hide most menu items.

The Properties of the ContextMenu Class

builtInItems
builtInItems
Returns an array of the built-in menu items.

customItems
customItems
An undefined array to contain menu item names.

onSelect
onSelect
Called before the menu is displayed.

ContextMenu Class Conformance

Class Name	Conformance
copy()	This method is unimplemented.
hideBuiltInItems()	This method is unimplemented.
builtInItems	This property is unimplemented.

Class Name	Conformance
customItems	This property is unimplemented.
onSelect	This property is unimplemented.

Math ActionScript Class

This class implements an Math object.

The Methods of the Math Class

abs()	abs()
acos()	acos()
asin()	asin()
atan()	atan()
atan2()	atan2()
ceil()	ceil()
cos()	cos()
exp()	exp()
floor()	floor()
log()	log()
max()	max()
min()	min()
pow()	pow()
random()	random()
round()	round()
sin()	sin()
sqrt()	sqrt()
tan()	tan()

The Properties of the Math Class

E	E

LN2	LN2
LN10	LN10
LOG2E	LOG2E
LOG10E	LOG10E
PI	PI
SQRT1_2	SQRT1_2
SQRT2	SQRT2

Math Class Conformance

Class Name	Conformance
abs()	This method is implemented.
acos()	This method is implemented.
asin()	This method is implemented.
atan()	This method is implemented.
atan2()	This method is implemented.
ceil()	This method is implemented.
cos()	This method is implemented.
exp()	This method is implemented.
floor()	This method is implemented.
log()	This method is implemented.
max()	This method is implemented.
min()	This method is implemented.
pow()	This method is implemented.
random()	This method is implemented.
round()	This method is implemented.
sin()	This method is implemented.

Class Name	Conformance
sqrt()	This method is implemented.
tan()	This method is implemented.
E	This property is implemented.
LN2	This property is implemented.
LN10	This property is implemented.
LOG2E	This property is implemented.
LOG10E	This property is implemented.
PI	This property is implemented.
SQRT1_2	This property is implemented.
SQRT2	This property is implemented.

NetConnection ActionScript Class

This class implements an NetConnection object. The NetConnection object opens a client connection to load an FLV video file from a local system, or a remote one using the RTMP or RTMPT protocols. RTMP connections use port 1935, and RTMPT connections use port 80.

This class was first introduced for swf v7 and the behaviour has recently changed slightly to allow the loading of files from external systems. Prior to this, you were required to pass NULL to this class. Gnash handles both cases. The older behaviour is default when connecting to the local filesystem for the file.

The URL that can be passed to *connect()* in newer versions of the swf format is as follows:

```
protocol:[//host][:port]/appname/[instanceName]
```

For protocol, specify either *RTMP* or *RTMPT*. If rtmp is specified, Flash Player will create a client connection. If *RTMPT* is specified, Flash Player will create an HTTP "tunneling" connection to the server.

You can omit the host parameter if the Flash movie is served from localhost. If the instanceName parameter is omitted, Flash Player connects to the application's default instance (_definst_).

The Methods of the Class

NetConnection.connect(const char *) the same host where Flash Communication Server is installed NetConnection.connect(const char *)
Connect to a local or remote system to load a FLV file. Arg can be NULL, or a URL.

NetConnection Class Conformance

Class Name	Conformance
NetConnection.connect()	This method is partially implemented. It can only be used in conjunction with the NetStream and Video classes; it cannot load a FLV file by itself.

Number ActionScript Class

This class implements an Number object.

The Methods of the Number Class

toString()	toString()
valueOf()	valueOf()

The Properties of the Number Class

MAX_VALUE	MAX_VALUE
MIN_VALUE	MIN_VALUE
NaN	NaN
NEGATIVE_INFINITY	NEGATIVE_INFINITY
POSITIVE_INFINITY	POSITIVE_INFINITY

Number Class Conformance

Class Name	Conformance
MAX_VALUE	Unimplemented.
MIN_VALUE	Unimplemented.
NaN	Implemented.
NEGATIVE_INFINITY	Implemented.
POSITIVE_INFINITY	Implemented.
toString()	Implemented.

Class Name	Conformance
valueOf()	Implemented.

String ActionScript Class

This class implements an String object.

The Methods of the String Class

charAt()	charAt()
charCodeAt()	charCodeAt()
concat()	concat()
fromCharCode()	fromCharCode()
indexOf()	indexOf()
lastIndexOf()	lastIndexOf()
slice()	slice()
split()	split()
substr()	substr()
substring()	substring()
toLowerCase()	toLowerCase()
toUpperCase()	toUpperCase()

The Properties of the String Class

length	length The length of the string.

String Class Conformance

Class Name	Conformance
charAt()	This method is implemented.
charCodeAt()	This method is implemented.
concat()	This method is not implemented.

Class Name	Conformance
fromCharCode()	This method is implemented.
indexOf()	This method is implemented.
lastIndexOf()	This method is not implemented.
slice()	This method is not implemented.
split()	This method is not implemented.
substr()	This method is not implemented.
substring()	This method is implemented.
toLowerCase()	The method is implemented.
toUpperCase()	This method is implemented.
length	This property is implemented.

CustomActions ActionScript Class

This class implements an CustomActions object.

The Methods of the Class

get()	get()
install()	install()
list()	list()
uninstall()	uninstall()

The Properties of the CustomActions Class

CustomActions Class Conformance

Class Name	Conformance
get()	This method is unimplemented.
install()	This method is unimplemented.

Class Name	Conformance
list()	This method is unimplemented.
uninstall()	This method is unimplemented.

Date ActionScript Class

This class implements an Date object.

The Methods of the Class

getDate()
getDay()
getFullYear()
getHours()
getMilliseconds()
getMinutes()
getMonth()
getSeconds()
getTime()
getTimezoneOffset()
getUTCDate()
getUTCDay()
getUTCFullYear()
getUTCHours()
getUTCMilliseconds()
getUTCMinutes()
getUTCMonth()
getUTCSeconds()
getYear()
setDate()
setFullYear()

getDate()
getDay()
getFullYear()
getHours()
getMilliseconds()
getMinutes()
getMonth()
getSeconds()
getTime()
getTimezoneOffset()
getUTCDate()
getUTCDay()
getUTCFullYear()
getUTCHours()
getUTCMilliseconds()
getUTCMinutes()
getUTCMonth()
getUTCSeconds()
getYear()
setDate()
setFullYear()

setHours()	setHours()
setMilliseconds()	setMilliseconds()
setMinutes()	setMinutes()
setMonth()	setMonth()
setSeconds()	setSeconds()
setTime()	setTime()
setUTCDate()	setUTCDate()
setUTCFullYear()	setUTCFullYear()
setUTCHours()	setUTCHours()
setUTCMilliseconds()	setUTCMilliseconds()
setUTCMinutes()	setUTCMinutes()
setUTCMonth()	setUTCMonth()
setUTCSeconds()	setUTCSeconds()
setYear()	setYear()
toString()	toString()
UTC()	UTC()

Date Class Conformance

Class Name	Conformance
getDate()	This method is unimplemented.
getDay()	This method is unimplemented.
getFullYear()	This method is unimplemented.
getHours()	This method is unimplemented.
getMilliseconds()	This method is unimplemented.
getMinutes()	This method is unimplemented.
getMonth()	This method is unimplemented.
getSeconds()	This method is unimplemented.
getTime()	This method is unimplemented.

Class Name	**Conformance**
getTimezoneOffset()	This method is unimplemented.
getUTCDate()	This method is unimplemented.
getUTCDay()	This method is unimplemented.
getUTCFullYear()	This method is unimplemented.
getUTCHours()	This method is unimplemented.
getUTCMilliseconds()	This method is unimplemented.
getUTCMinutes()	This method is unimplemented.
getUTCMonth()	This method is unimplemented.
getUTCSeconds()	This method is unimplemented.
getYear()	This method is unimplemented.
setDate()	This method is unimplemented.
setFullYear()	This method is unimplemented.
setHours()	This method is unimplemented.
setMilliseconds()	This method is unimplemented.
setMinutes()	This method is unimplemented.
setMonth()	This method is unimplemented.
setSeconds()	This method is unimplemented.
setTime()	This method is unimplemented.
setUTCDate()	This method is unimplemented.
setUTCFullYear()	This method is unimplemented.
setUTCHours()	This method is unimplemented.
setUTCMilliseconds()	This method is unimplemented.
setUTCMinutes()	This method is unimplemented.
setUTCMonth()	This method is unimplemented.

Class Name	Conformance
setUTCSeconds()	This method is unimplemented.
setYear()	This method is unimplemented.
toString()	This method is unimplemented.
UTC()	This method is unimplemented.

Error ActionScript Class

This class implements an Error object.

The Methods of the Class

toString() toString()
Returns the error object as a string

The Properties of the Error Class

message message
Returns an error message as a string.

name name
Returns the name of the error object.

Error Class Conformance

Class Name	Conformance
toString()	This method is unimplemented.
message	This property is unimplemented.
name	This property is unimplemented.

Function ActionScript Class

This class implements an Function object.

The Methods of the Class

apply() apply()

call() call()

The Properties of the Function Class

Function Class Conformance

Class Name	Conformance
apply()	This method has an unknown status.
call()	This method has an unknown status.

Key ActionScript Class

This class implements an Key object.

The Methods of the Class

addListener() addListener()
Add a callback for key up and down events.

getAscii() getAscii()
Return the ACSII value of the key.

getCode() getCode()
Return the key code of the key.

isDown() isDown()
Returns true if the specified key is down.

isToggled() isToggled()
Returns true if the Num Lock key is on.

removeListener() removeListener()
Remove the event handler.

The Properties of the Key Class

BACKSPACE BACKSPACE
A constant representing the backspace key.

CAPSLOCK CAPSLOCK
A constant representing the Caps Lock key.

CONTROL CONTROL

	A constant representing the Control key.
DELETEKEY	DELETEKEY A constant representing the Delete key.
DOWN	DOWN A constant representing the Down arrow key.
END	END A constant representing the End key.
ENTER	ENTER A constant representing the Enter key.
ESCAPE	ESCAPE A constant representing the Escape key.
HOME	HOME A constant representing the Home key.
INSERT	INSERT A constant representing the Insert key.
LEFT	LEFT A constant representing the left arrow key.
PGDN	PGDN A constant representing the Page Down key.
PGUP	PGUP A constant representing the Page Up key.
RIGHT	RIGHT A constant representing the right arrow key.
SHIFT	SHIFT A constant representing the shift key.
SPACE	SPACE
TAB	TAB A constant representing the space bar.
UP	UP A constant representing the up arrow key.

Key Class Event Handlers

onKeyDown	onKeyDown
onKeyUp	onKeyUp

Key Class Conformance

Class Name	**Conformance**
addListener()	This method is implemented.
getAscii()	This method is implemented.
getCode()	This method is implemented.
isDown()	This method is implemented.
isToggled()	This method is implemented.
removeListener()	This method is implemented.
BACKSPACE	This constant is implemented.
CAPSLOCK	This constant is implemented.
CONTROL	This constant is implemented.
DELETEKEY	This constant is implemented.
DOWN	This constant is implemented.
END	This constant is implemented.
ENTER	This constant is implemented.
ESCAPE	This constant is implemented.
HOME	This constant is implemented.
INSERT	This constant is implemented.
LEFT	This constant is implemented.
onKeyDown	This constant is implemented.
onKeyUp	This constant is implemented.
PGDN	This constant is implemented.
PGUP	This constant is implemented.
RIGHT	This constant is implemented.
SHIFT	This constant is implemented.
SPACE	This constant is implemented.

Class Name	Conformance
TAB	This constant is implemented.
UP	This constant is implemented.

LoadVars ActionScript Class

This class implements an LoadVars object.

The Methods of the Class

addRequestHeader()	addRequestHeader()
decode()	decode()
getBytesLoaded()	getBytesLoaded()
getBytesTotal()	getBytesTotal()
load()	load()
send()	send()
sendAndLoad()	sendAndLoad()
toString()	toString()

The Properties of the LoadVars Class

contentType	contentType
loaded	loaded
onData	onData
onLoad	onLoad

LoadVars Class Conformance

Class Name	Conformance
addRequestHeader()	This method is unimplemented.
decode()	This method is unimplemented.
getBytesLoaded()	This method is unimplemented.

Class Name	Conformance
getBytesTotal()	This method is unimplemented.
load()	This method is unimplemented.
send()	This method is unimplemented.
sendAndLoad()	This method is unimplemented.
toString()	This method is unimplemented.
contentType	This property is unimplemented.
loaded	This property is unimplemented.
onData	This property is unimplemented.
onLoad	This property is unimplemented.

LocalConnection ActionScript Class

This class implements an LocalConnection object. This is basically a Flash version of the a standard RPC (Remote Procedure Call). This is used to have multiple flash movies exchange data without converting it to *XML* first. This class uses shared memory to transfer the data in the *AMF* format.

There is some documentation of the *AMF* format online, but not much. There appear to be a few free *AMF* projects, but none that I could find have any code. So I'm stuck developing a C/C++ based *AMF* library to transfer data.

The Methods of the Class

close()
close()
Close the current open connection, and remove the shared memory segment.

connect(const char *)
connect(const char *)
This establishes a connection point so other movies can remotely execute methods in this movie. The shared memory segment is created in a form that can be attached to by a client. The code that actually remotely executes a method has yet to be written for Gnash.

domain()
domain()
This returns the network domain for this connection. The default is to return "localhost".

send()
send()
Execute a method of the local object. Data is encoded in the *AMF* format.

The Event Handlers of the LocalConnection Class

allowDomain	allowDomain Called for every method being executed.
allowInsecureDomain	allowInsecureDomain Called when a request is received over the network.
onStatus	onStatus Returns the status of the method that was executed.

LocalConnection Class Conformance

Class Name	**Conformance**
close()	This method is implemented.
connect()	This method is implemented.
domain()	This method is implemented.
send()	This method is unimplemented.
allowDomain	This property is unimplemented.
allowInsecureDomain	This property is unimplemented.
onStatus	This property is unimplemented.

Microphone ActionScript Class

This class implements an Microphone object.

The Methods of the Class

get()	get()
setGain()	setGain()
setRate()	setRate()
setSilenceLevel()	setSilenceLevel()
setUseEchoSuppression()	setUseEchoSuppression()

The Properties of the Microphone Class

activityLevel	activityLevel
gain	gain
index	index
muted	muted
name	name
names	names
onActivity	onActivity
onStatus	onStatus
rate	rate
silenceLevel	silenceLevel
silenceTimeOut	silenceTimeOut
useEchoSuppression	useEchoSuppression

Microphone Class Conformance

Class Name	**Conformance**
get()	This method is unimplemented.
setGain()	This method is unimplemented.
setRate()	This method is unimplemented.
setSilenceLevel()	This method is unimplemented.
setUseEchoSuppression()	This method is unimplemented.
activityLevel	This property is unimplemented.
gain	This property is unimplemented.
index	This property is unimplemented.
muted	This property is unimplemented.
name	This property is unimplemented.
names	This property is unimplemented.
onActivity	This property is unimplemented.

Class Name	Conformance
onStatus	This property has an unknown status.
rate	This property has an unknown status.
silenceLevel	This property has an unknown status.
silenceTimeOut	This property has an unknown status.
useEchoSuppression	This property has an unknown status.

Mouse ActionScript Class

This class implements an Mouse object.

The Methods of the Class

addListener()	addListener()
hide()	hide()
removeListener()	removeListener()
show()	show()

The Properties of the Mouse Class

onMouseDown	onMouseDown
onMouseMove	onMouseMove
onMouseUp	onMouseUp
onMouseWheel	onMouseWheel

Mouse Class Conformance

Class Name	Conformance
addListener()	This method is unimplemented.
hide()	This method is unimplemented.
removeListener()	This method is unimplemented.

Class Name	Conformance
show()	This method is unimplemented.
onMouseDown	This property is unimplemented.
onMouseMove	This property is unimplemented.
onMouseUp	This property is unimplemented.
onMouseWheel	This property is unimplemented.

MovieClip ActionScript Class

This class implements an MovieClip object.

The Methods of the Class

attachAudio()

attachMovie()

beginFill()

beginGradientFill()

clear()

createEmptyMovieClip()

createTextField()

curveTo()

duplicateMovieClip()

endFill()

getBounds()

getBytesLoaded()

getBytesTotal()

getDepth()

getInstanceAtDepth()

getNextHighestDepth()

getSWFVersion()

getTextSnapshot()

attachAudio()

attachMovie()

beginFill()

beginGradientFill()

clear()

createEmptyMovieClip()

createTextField()

curveTo()

duplicateMovieClip()

endFill()

getBounds()

getBytesLoaded()

getBytesTotal()

getDepth()

getInstanceAtDepth()

getNextHighestDepth()

getSWFVersion()

getTextSnapshot()

getURL()	getURL()
globalToLocal()	globalToLocal()
gotoAndPlay()	gotoAndPlay()
gotoAndStop()	gotoAndStop()
hitTest()	hitTest()
lineStyle()	lineStyle()
lineTo()	lineTo()
loadMovie()	loadMovie()
loadVariables()	loadVariables()
localToGlobal()	localToGlobal()
moveTo()	moveTo()
nextFrame()	nextFrame()
play()	play()
prevFrame()	prevFrame()
removeMovieClip()	removeMovieClip()
setMask()	setMask()
startDrag()	startDrag()
stop()	stop()
stopDrag()	stopDrag()
swapDepths()	swapDepths()
unloadMovie()	unloadMovie()

The Properties of the MovieClip Class

_alpha	_alpha
_currentframe	_currentframe
_droptarget	_droptarget
enabled	enabled
focusEnabled	focusEnabled
_focusrect	_focusrect
_framesloaded	_framesloaded

_height	_height
hitArea	hitArea
_lockroot	_lockroot
menu	menu
_name	_name
onData	onData
onDragOut	onDragOut
onDragOver	onDragOver
onEnterFrame	onEnterFrame
onKeyDown	onKeyDown
onKeyUp	onKeyUp
onKillFocus	onKillFocus
onLoad	onLoad
onMouseDown	onMouseDown
onMouseMove	onMouseMove
onMouseUp	onMouseUp
onPress	onPress
onRelease	onRelease
onReleaseOutside	onReleaseOutside
onRollOut	onRollOut
onRollOver	onRollOver
onSetFocus	onSetFocus
onUnload	onUnload
_parent	_parent
_quality	_quality
_rotation	_rotation
_soundbuftime	_soundbuftime
tabChildren	tabChildren
tabEnabled	tabEnabled
tabIndex	tabIndex

_target	_target
_totalframes	_totalframes
trackAsMenu	trackAsMenu
_url	_url
useHandCursor	useHandCursor
_visible	_visible
_width	_width
_x	_x
_xmouse	_xmouse
_xscale	_xscale
_y	_y
_ymouse	_ymouse
_yscale	_yscale

MovieClip Class Conformance

Class Name	Conformance
attachAudio()	This method has an unknown status.
attachMovie()	This method has an unknown status.
beginFill()	This method has an unknown status.
beginGradientFill()	This method has an unknown status.
clear()	This method has an unknown status.
createEmptyMovieClip()	This method has an unknown status.
createTextField()	This method has an unknown status.
curveTo()	This method has an unknown status.
duplicateMovieClip()	This method has an unknown status.
endFill()	This method has an unknown status.
getBounds()	This method has an unknown status.

Class Name	Conformance
getBytesLoaded()	This method has an unknown status.
getBytesTotal()	This method has an unknown status.
getDepth()	This method has an unknown status.
getInstanceAtDepth()	This method has an unknown status.
getNextHighestDepth()	This method has an unknown status.
getSWFVersion()	This method has an unknown status.
getTextSnapshot()	This method has an unknown status.
getURL()	This method has an unknown status.
globalToLocal()	This method has an unknown status.
gotoAndPlay()	This method has an unknown status.
gotoAndStop()	This method has an unknown status.
hitTest()	This method has an unknown status.
lineStyle()	This method has an unknown status.
lineTo()	This method has an unknown status.
loadMovie()	This method has an unknown status.
loadVariables()	This method has an unknown status.
localToGlobal()	This method has an unknown status.
moveTo()	This method has an unknown status.
nextFrame()	This method has an unknown status.
play()	This method has an unknown status.
prevFrame()	This method has an unknown status.
removeMovieClip()	This method has an unknown status.
setMask()	This method has an unknown status.
startDrag()	This method has an unknown status.

Class Name	Conformance
stop()	This method has an unknown status.
stopDrag()	This method has an unknown status.
swapDepths()	This method has an unknown status.
unloadMovie()	This method has an unknown status.
_alpha	This property has an unknown status.
_currentframe	This property has an unknown status.
_droptarget	This property has an unknown status.
enabled	This property has an unknown status.
focusEnabled	This property has an unknown status.
_focusrect	This property has an unknown status.
_framesloaded	This property has an unknown status.
_height	This property has an unknown status.
hitArea	This property has an unknown status.
_lockroot	This property has an unknown status.
menu	This property has an unknown status.
_name	This property has an unknown status.
onData	This property has an unknown status.
onDragOut	This property has an unknown status.
onDragOver	This property has an unknown status.
onEnterFrame	This property has an unknown status.
onKeyDown	This property has an unknown status.
onKeyUp	This property has an unknown status.
onKillFocus	This property has an unknown status.
onLoad	This property has an unknown status.

Class Name	Conformance
onMouseDown	This property has an unknown status.
onMouseMove	This property has an unknown status.
onMouseUp	This property has an unknown status.
onPress	This property has an unknown status.
onRelease	This property has an unknown status.
onReleaseOutside	This property has an unknown status.
onRollOut	This property has an unknown status.
onRollOver	This property has an unknown status.
onSetFocus	This property has an unknown status.
onUnload	This property has an unknown status.
_parent	This property has an unknown status.
_quality	This property has an unknown status.
_rotation	This property has an unknown status.
_soundbuftime	This property has an unknown status.
tabChildren	This property has an unknown status.
tabEnabled	This property has an unknown status.
tabIndex	This property has an unknown status.
_target	This property has an unknown status.
_totalframes	This property has an unknown status.
trackAsMenu	This property has an unknown status.
_url	This property has an unknown status.
useHandCursor	This property has an unknown status.
_visible	This property has an unknown status.
_width	This property has an unknown status.

Class Name	Conformance
_x	This property has an unknown status.
_xmouse	This property has an unknown status.
_xscale	This property has an unknown status.
_y	This property has an unknown status.
_ymouse	This property has an unknown status.
_yscale	This property has an unknown status.

MovieClipLoader ActionScript Class

This class implements an MovieClipLoader object.

The Methods of the Class

addListener()	addListener()
getProgress()	getProgress()
loadClip()	loadClip()
removeListener()	removeListener()
unloadClip()	unloadClip()

The Properties of the MovieClipLoader Class

onLoadComplete	onLoadComplete
onLoadError	onLoadError
onLoadInit	onLoadInit
onLoadProgress	onLoadProgress
onLoadStart	onLoadStart

MovieClipLoader Class Conformance

Class Name	Conformance
addListener()	This method has an unknown status.

Class Name	Conformance
getProgress()	This method has an unknown status.
loadClip()	This method has an unknown status.
removeListener()	This method has an unknown status.
unloadClip()	This method has an unknown status.
onLoadComplete	This property has an unknown status.
onLoadError	This property has an unknown status.
onLoadInit	This property has an unknown status.
onLoadProgress	This property has an unknown status.
onLoadStart	This property has an unknown status.

NetStream ActionScript Class

This class implements an NetStream object.

The Methods of the Class

close()	close()
pause()	pause()
play()	play()
seek()	seek()
setBufferTime()	setBufferTime()

The Properties of the NetStream Class

bufferLength	bufferLength
bufferTime	bufferTime
bytesLoaded	bytesLoaded
bytesTotal	bytesTotal
currentFps	currentFps
onStatus	onStatus

time time

NetStream Class Conformance

Class Name	Conformance
close()	This method is unimplemented.
pause()	This method is unimplemented.
play()	This method is unimplemented.
seek()	This method is unimplemented.
setBufferTime()	This method is unimplemented.
bufferLength	This property is unimplemented.
bufferTime	This property is unimplemented.
bytesLoaded	This property is unimplemented.
bytesTotal	This property is unimplemented.
currentFps	This property is unimplemented.
onStatus	This property is unimplemented.
time	This property is unimplemented.

Object ActionScript Class

This class implements an Object object.

The Methods of the Class

addProperty() addProperty()

registerClass() registerClass()

toString() toString()

unwatch() unwatch()

valueOf() valueOf()

watch() watch()

Sharedclear()	Sharedclear()
Sharedflush()	Sharedflush()
SharedgetLocal()	SharedgetLocal()
SharedgetSize()	SharedgetSize()

The Properties of the Object Class

constructor	constructor
__proto__	__proto__
__resolve	__resolve
Shareddata	Shareddata
SharedonStatus	SharedonStatus

Object Class Conformance

Class Name	**Conformance**
addProperty()	This method has an unknown status.
registerClass()	This method has an unknown status.
toString()	This method has an unknown status.
unwatch()	This method has an unknown status.
valueOf()	This method has an unknown status.
watch()	This method has an unknown status.
Sharedclear()	This method has an unknown status.
Sharedflush()	This method has an unknown status.
SharedgetLocal()	This method has an unknown status.
SharedgetSize()	This method has an unknown status.
constructor	This property has an unknown status.
__proto__	This property has an unknown status.
__resolve	This property has an unknown status.

Class Name	Conformance
Shareddata	This property has an unknown status.
SharedonStatus	This property has an unknown status.

Selection ActionScript Class

This class implements an Selection object.

The Methods of the Class

addListener()	addListener()
getBeginIndex()	getBeginIndex()
getCaretIndex()	getCaretIndex()
getEndIndex()	getEndIndex()
getFocus()	getFocus()
removeListener()	removeListener()
setFocus()	setFocus()
setSelection()	setSelection()

The Properties of the Selection Class

onSetFocus	onSetFocus

Selection Class Conformance

Class Name	Conformance
addListener()	This method is unimplemented.
getBeginIndex()	This method is unimplemented.
getCaretIndex()	This method is unimplemented.
getEndIndex()	This method is unimplemented.
getFocus()	This method is unimplemented.

Class Name	Conformance
removeListener()	This method is unimplemented.
setFocus()	This method is unimplemented.
setSelection()	This method is unimplemented.
onSetFocus	This property is unimplemented.

SharedObject ActionScript Class

This class implements an SharedObject object.

The Methods of the Class

clear()	clear()
flush()	flush()
getLocal()	getLocal()
getSize()	getSize()

The Properties of the SharedObject Class

data	data
onStatus	onStatus

SharedObject Class Conformance

Class Name	Conformance
clear()	This method is unimplemented.
flush()	This method is unimplemented.
getLocal()	This method is unimplemented.
getSize()	This method is unimplemented.
data	This property is unimplemented.
onStatus	This property is unimplemented.

Sound ActionScript Class

This class implements an Sound object.

The Methods of the Class

attachSound()	attachSound() Attach the specified sound.
getBytesLoaded()	getBytesLoaded() Returns how many bytes of the sound have been loaded.
getBytesTotal()	getBytesTotal() Returns the total size of the sound.
getPan()	getPan() Returns the value of the previous call to setPan().
getTransform()	getTransform() Returns the value of the previous call to setTransform().
getVolume()	getVolume() Returns the value of the previous call to setVolumne().
loadSound()	loadSound() Load an MP3 file.
setPan()	setPan() Change the balance.
setTransform()	setTransform() Set the channel amounts for each speaker.
setVolume()	setVolume() Set the volumne for playing.
start()	start() Play the loaded sound.
stop()	stop() Stop playing the sound.

The Properties of the Sound Class

duration	duration The length in milliseconds of the sound.
id3	id3 Returns the ID3 tag of an MP3 file.
position	position Returns the amount of time the sound has been playing.

Event Handlers of the Sound Class

onID3	onID3 Called when ID3 data is available.
onLoad	onLoad Called when a sound is loaded.
onSoundComplete	onSoundComplete Called when a sound is done playing.

Sound Class Conformance

Class Name	**Conformance**
attachSound()	This method is implemented.
getBytesLoaded()	This method has an unknown status.
getBytesTotal()	This method has an unknown status.
getPan()	This method is unimplemented.
getTransform()	This method has an unknown status.
getVolume()	This method is unimplemented.
loadSound()	This method is unimplemented.
setPan()	This method is unimplemented.
setTransform()	This method is unimplemented.
setVolume()	This method is unimplemented.
start()	This method is implemented.
stop()	This method is implemented.
duration	This property has an unknown status.
id3	This property is unimplemented.
onID3	This event handler is unimplemented.
onLoad	This event handler is unimplemented.
onSoundComplete	This event handler is unimplemented.

Class Name	Conformance
position	This property is implemented.

Stage ActionScript Class

This class implements an Stage object.

The Methods of the Class

addListener()	addListener()
removeListener()	removeListener()

The Properties of the Stage Class

align	align
height	height
onResize	onResize
scaleMode	scaleMode
showMenu	showMenu
width	width

Stage Class Conformance

Class Name	Conformance
addListener()	This method is unimplemented.
removeListener()	This method is unimplemented.
align	This property is unimplemented.
height	This property is unimplemented.
onResize	This property is unimplemented.
scaleMode	This property is unimplemented.
showMenu	This property is unimplemented.

Class Name	Conformance
width	This property is unimplemented.

System ActionScript Class

This class implements an System object.

The Methods of the Class

security.allowDomain()	security.allowDomain()
security.allowInsecureDomain()	security.allowInsecureDomain()
security.loadPolicyFile()	security.loadPolicyFile()
setClipboard()	setClipboard()
showSettings()	showSettings()

The Properties of the System Class

capabilities	capabilities
object	object
capabilities.avHardwareDisable	capabilities.avHardwareDisable
capabilities.hasAccessibility	capabilities.hasAccessibility
capabilities.hasAudio	capabilities.hasAudio
capabilities.hasAudioEncoder	capabilities.hasAudioEncoder
capabilities.hasEmbeddedVideo	capabilities.hasEmbeddedVideo
capabilities.hasMP3	capabilities.hasMP3
capabilities.hasPrinting	capabilities.hasPrinting
capabilities.hasScreenBroadcast	capabilities.hasScreenBroadcast
capabilities.hasScreenPlayback	capabilities.hasScreenPlayback
capabilities.hasStreamingAudio	capabilities.hasStreamingAudio
capabilities.hasStreamingVideo	capabilities.hasStreamingVideo
capabilities.hasVideoEncoder	capabilities.hasVideoEncoder
capabilities.isDebugger	capabilities.isDebugger

capabilities.language	capabilities.language
capabilities.localFileReadDisable	capabilities.localFileReadDisable
capabilities.manufacturer	capabilities.manufacturer
capabilities.os	capabilities.os
capabilities.pixelAspectRatio	capabilities.pixelAspectRatio
capabilities.playerType	capabilities.playerType
capabilities.screenColor	capabilities.screenColor
capabilities.screenDPI	capabilities.screenDPI
capabilities.screenResolutionX	capabilities.screenResolutionX
capabilities.screenResolutionY	capabilities.screenResolutionY
capabilities.serverString	capabilities.serverString
capabilities.version	capabilities.version
security	security
object	object
exactSettings	exactSettings
onStatus	onStatus
useCodepage	useCodepage

System Class Conformance

Class Name	**Conformance**
security.allowDomain()	This method has an unknown status.
security.allowInsecureDomain()	This method has an unknown status.
security.loadPolicyFile()	This method has an unknown status.
setClipboard()	This method has an unknown status.
showSettings()	This method has an unknown status.
capabilities	This property has an unknown status.
object	This property has an unknown status.
capabilities.avHardwareDisable	This property has an unknown status.

Class Name	Conformance
capabilities.hasAccessibility	This property has an unknown status.
capabilities.hasAudio	This property has an unknown status.
capabilities.hasAudioEncoder	This property has an unknown status.
capabilities.hasEmbeddedVideo	This property has an unknown status.
capabilities.hasMP3	This property has an unknown status.
capabilities.hasPrinting	This property has an unknown status.
capabilities.hasScreenBroadcast	This property has an unknown status.
capabilities.hasScreenPlayback	This property has an unknown status.
capabilities.hasStreamingAudio	This property has an unknown status.
capabilities.hasStreamingVideo	This property has an unknown status.
capabilities.hasVideoEncoder	This property has an unknown status.
capabilities.isDebugger	This property has an unknown status.
capabilities.language	This property has an unknown status.
capabilities.localFileReadDisable	This property has an unknown status.
capabilities.manufacturer	This property has an unknown status.
capabilities.os	This property has an unknown status.
capabilities.pixelAspectRatio	This property has an unknown status.
capabilities.playerType	This property has an unknown status.
capabilities.screenColor	This property has an unknown status.
capabilities.screenDPI	This property has an unknown status.
capabilities.screenResolutionX	This property has an unknown status.
capabilities.screenResolutionY	This property has an unknown status.
capabilities.serverString	This property has an unknown status.
capabilities.version	This property has an unknown status.

Class Name	Conformance
security	This property has an unknown status.
object	This property has an unknown status.
exactSettings	This property has an unknown status.
onStatus	This property has an unknown status.
useCodepage	This property has an unknown status.

TextField ActionScript Class

This class implements an TextField object.

The Methods of the Class

StyleSheet.clear()

StyleSheet.getStyle()

StyleSheet.getStyleNames()

StyleSheet.load()

StyleSheet.parseCSS()

StyleSheet.setStyle()

StyleSheet.transform()

addListener()

getDepth()

getFontList()

getNewTextFormat()

getTextFormat()

removeListener()

removeTextField()

replaceSel()

replaceText()

setNewTextFormat()

setTextFormat()

StyleSheet.clear()

StyleSheet.getStyle()

StyleSheet.getStyleNames()

StyleSheet.load()

StyleSheet.parseCSS()

StyleSheet.setStyle()

StyleSheet.transform()

addListener()

getDepth()

getFontList()

getNewTextFormat()

getTextFormat()

removeListener()

removeTextField()

replaceSel()

replaceText()

setNewTextFormat()

setTextFormat()

The Properties of the TextField Class

StyleSheet	StyleSheet
class	class
StyleSheet.onLoad	StyleSheet.onLoad
_alpha	_alpha
autoSize	autoSize
background	background
backgroundColor	backgroundColor
border	border
borderColor	borderColor
bottomScroll	bottomScroll
condenseWhite	condenseWhite
embedFonts	embedFonts
_height	_height
hscroll	hscroll
html	html
htmlText	htmlText
length	length
maxChars	maxChars
maxhscroll	maxhscroll
maxscroll	maxscroll
menu	menu
mouseWheelEnabled	mouseWheelEnabled
multiline	multiline
_name	_name
onChanged	onChanged
onKillFocus	onKillFocus
onScroller	onScroller

onSetFocus	onSetFocus
_parent	_parent
password	password
_quality	_quality
restrict	restrict
_rotation	_rotation
scroll	scroll
selectable	selectable
styleSheet	styleSheet
tabEnabled	tabEnabled
tabIndex	tabIndex
_target	_target
text	text
textColor	textColor
textHeight	textHeight
textWidth	textWidth
type	type
_url	_url
variable	variable
_visible	_visible
_width	_width
wordWrap	wordWrap
_x	_x
_xmouse	_xmouse
_xscale	_xscale
_y	_y
_ymouse	_ymouse
_yscale	_yscale

TextField Class Conformance

Class Name	Conformance
StyleSheet.clear()	This method has an unknown status.
StyleSheet.getStyle()	This method has an unknown status.
StyleSheet.getStyleNames()	This method has an unknown status.
StyleSheet.load()	This method has an unknown status.
StyleSheet.parseCSS()	This method has an unknown status.
StyleSheet.setStyle()	This method has an unknown status.
StyleSheet.transform()	This method has an unknown status.
addListener()	This method has an unknown status.
getDepth()	This method has an unknown status.
getFontList()	This method has an unknown status.
getNewTextFormat()	This method has an unknown status.
getTextFormat()	This method has an unknown status.
removeListener()	This method has an unknown status.
removeTextField()	This method has an unknown status.
replaceSel()	This method has an unknown status.
replaceText()	This method has an unknown status.
setNewTextFormat()	This method has an unknown status.
setTextFormat()	This method has an unknown status.
StyleSheet	This property has an unknown status.
class	This property has an unknown status.
StyleSheet.onLoad	This property has an unknown status.
_alpha	This property has an unknown status.
autoSize	This property has an unknown status.
background	This property has an unknown status.

Class Name	Conformance
backgroundColor	This property has an unknown status.
border	This property has an unknown status.
borderColor	This property has an unknown status.
bottomScroll	This property has an unknown status.
condenseWhite	This property has an unknown status.
embedFonts	This property has an unknown status.
_height	This property has an unknown status.
hscroll	This property has an unknown status.
html	This property has an unknown status.
htmlText	This property has an unknown status.
length	This property has an unknown status.
maxChars	This property has an unknown status.
maxhscroll	This property has an unknown status.
maxscroll	This property has an unknown status.
menu	This property has an unknown status.
mouseWheelEnabled	This property has an unknown status.
multiline	This property has an unknown status.
_name	This property has an unknown status.
onChanged	This property has an unknown status.
onKillFocus	This property has an unknown status.
onScroller	This property has an unknown status.
onSetFocus	This property has an unknown status.
_parent	This property has an unknown status.
password	This property has an unknown status.

Class Name	Conformance
_quality	This property has an unknown status.
restrict	This property has an unknown status.
_rotation	This property has an unknown status.
scroll	This property has an unknown status.
selectable	This property has an unknown status.
styleSheet	This property has an unknown status.
tabEnabled	This property has an unknown status.
tabIndex	This property has an unknown status.
_target	This property has an unknown status.
text	This property has an unknown status.
textColor	This property has an unknown status.
textHeight	This property has an unknown status.
textWidth	This property has an unknown status.
type	This property has an unknown status.
_url	This property has an unknown status.
variable	This property has an unknown status.
_visible	This property has an unknown status.
_width	This property has an unknown status.
wordWrap	This property has an unknown status.
_x	This property has an unknown status.
_xmouse	This property has an unknown status.
_xscale	This property has an unknown status.
_y	This property has an unknown status.
_ymouse	This property has an unknown status.

Class Name	Conformance
_yscale	This property has an unknown status.

TextFormat ActionScript Class

This class implements an TextFormat object.

The Methods of the Class

getTextExtent()	getTextExtent()

The Properties of the TextFormat Class

align	align
blockIndent	blockIndent
bold	bold
bullet	bullet
color	color
font	font
indent	indent
italic	italic
leadingxmlsocket.xml	leadingxmlsocket.xml
leftMargin	leftMargin
rightMargin	rightMargin
size	size
tabStops	tabStops
target	target
underline	underline
url	url

TextFormat Class Conformance

Class Name	Conformance
getTextExtent()	This method has an unknown status.
align	This property has an unknown status.
blockIndent	This property has an unknown status.
bold	This property has an unknown status.
bullet	This property has an unknown status.
color	This property has an unknown status.
font	This property has an unknown status.
indent	This property has an unknown status.
italic	This property has an unknown status.
leadingxmlsocket.xml	This property has an unknown status.
leftMargin	This property has an unknown status.
rightMargin	This property has an unknown status.
size	This property has an unknown status.
tabStops	This property has an unknown status.
target	This property has an unknown status.
underline	This property has an unknown status.
url	This property has an unknown status.

TextSnapshot ActionScript Class

This class implements an TextSnapshot object.

The Methods of the Class

findText()	findText()
getCount()	getCount()
getSelected()	getSelected()
getSelectedText()	getSelectedText()

getText()	getText()
hitTestTextNearPos()	hitTestTextNearPos()
setSelectColor()	setSelectColor()
setSelected()	setSelected()

The Properties of the TextSnapshot Class

TextSnapshot Class Conformance

Class Name	Conformance
findText()	This method is unimplemented.
getCount()	This method is unimplemented.
getSelected()	This method is unimplemented.
getSelectedText()	This method is unimplemented.
getText()	This method is unimplemented.
hitTestTextNearPos()	This method has an unknown status.
setSelectColor()	This method has an unknown status.
setSelected()	This method has an unknown status.

Video ActionScript Class

This class implements an Video object.

The Methods of the Class

attachVideo()	attachVideo()
clear()	clear()

The Properties of the Video Class

deblocking	deblocking

height	height
smoothing	smoothing
width	width

Video Class Conformance

Class Name	Conformance
attachVideo()	This method is unimplemented.
clear()	This method is unimplemented.
deblocking	This property is unimplemented.
height	This property is unimplemented.
smoothing	This property is unimplemented.
width	This property is unimplemented.

XMLNode ActionScript Class

This class implements an XMLNode object. This is mostly only used internally by Gnash for holding the data for a node. It does exist within the interpreter as a valid object though, so its properties and methods can be accessed by a flash movie script.

As the XML class is derived from this one, many of these methods are the same as for that class.

The Methods of the Class

appendChild()	appendChild() Append a child node to this node.
cloneNode()	cloneNode() Copy a node, returning an XMLNode *.
hasChildNodes()	hasChildNodes() Return true if this node has any children.
insertBefore()	insertBefore() Insert a node before this node.
removeNode()	removeNode() Remove a node from the parent's list.
toString()	toString() Convert the node and its children to a string representation.

The Properties of the XMLNode Class

attributes	attributes Returns an array of the attributes of a node.
childNodes	childNodes Returns an array of the children of a node.
firstChild	firstChild Returns the first child node.
lastChild	lastChild Returns the last child node.
nextSibling	nextSibling Returns the next child node.
previousSibling	previousSibling Returns the child node before this one.
nodeName	nodeName Returns the name of the node.
nodeType	nodeType Returns the type of a node.
nodeValue	nodeValue Contains the text for a text node type.
parentNode	parentNode Returns this node's parent.

XMLNode Class Conformance

Class Name	**Conformance**
appendChild()	This method is implemented.
cloneNode()	This method is implemented.
hasChildNodes()	This method is implemented.
insertBefore()	This method is unimplemented.
removeNode()	This method is unimplemented.
toString()	This method is unimplemented.
attributes	This property is unimplemented.
childNodes	This property is unimplemented.

Class Name	Conformance
firstChild	This property is implemented.
lastChild	This property is implemented.
nextSibling	This property is implemented.
previousSibling	This property is implemented.
nodeName	This property is implemented.
nodeType	This property is implemented.
nodeValue	This property is implemented.
parentNode	This property is unimplemented.

XML ActionScript Class

This class implements an XML object.

The Methods of the Class

addRequestHeader()
addRequestHeader()
Change the HTTP header.

appendChild()
appendChild()
Append a child node to this node.

cloneNode()
cloneNode()
Copy a node.

createElement()
createElement()
Create an element for a node.

createTextNode()
createTextNode()
Create a text node.

getBytesLoaded()
getBytesLoaded()

getBytesTotal()
getBytesTotal()
Return the size of the XML source.

hasChildNodes()
hasChildNodes()
Return if this node has any children.

insertBefore()
insertBefore()
Insert a node before this node.

load()
load()
Load a XML message in memory.

parseXML()	parseXML() Parse an XML document.
removeNode()	removeNode() Remove a node.
send()	send() Send the node through the network.
sendAndLoad()	sendAndLoad() Send a node, and get the result using the network.
toString()	toString() Convert the node and its children to a string.

The Properties of the XML Class

contentType	contentType The MIME type.
attributes	attributes Returns an array of the attributes of a node.
childNodes	childNodes Returns an array of the children of a node.
xmlDecl	xmlDecl Specify document's declaration.
docTypeDecl	docTypeDecl Get a string version of a document's declaration.
firstChild	firstChild Returns the first child node.
ignoreWhite	ignoreWhite If set, blank nodes are deleted.
lastChild	lastChild Returns the last child node.
loaded	loaded A flag that signifies if a file was loaded.
nextSibling	nextSibling Returns the next child node.
nodeName	nodeName Returns the name of the node.
nodeType	nodeType Returns the type of a node.
nodeValue	nodeValue Contains the text for a text node type.
parentNode	parentNode

	Returns this node's parent.
status	status Returns the status code from parsing the XML document.
previousSibling	previousSibling Returns the child node before this one.

XML Class Conformance

Class Name	Conformance
addRequestHeader()	This method is unimplemented.
appendChild()	This method is unimplemented.
cloneNode()	This method is unimplemented.
createElement()	This method is unimplemented.
createTextNode()	This method is unimplemented.
getBytesLoaded()	This method is unimplemented.
getBytesTotal()	This method is unimplemented.
hasChildNodes()	This method is implemented.
insertBefore()	This method is unimplemented.
load()	This method is implemented.
parseXML()	This method is implemented.
removeNode()	This method is unimplemented.
send()	This method is unimplemented.
sendAndLoad()	This method is unimplemented.
toString()	This method is unimplemented.
contentType	This property is unimplemented.
attributes	This property is implemented.
childNodes	This property is implemented.
xmlDecl	This property is unimplemented.

Class Name	Conformance
docTypeDecl	This property is unimplemented.
firstChild	This property is implemented.
ignoreWhite	This property is unimplemented.
lastChild	This property is unimplemented.
loaded	This property is implemented.
nextSibling	This property is unimplemented.
nodeName	This property is unimplemented.
nodeType	This property is implemented.
nodeValue	This property is unimplemented.
onData	This event handler is implemented.
onLoad	This event handler is implemented.
parentNode	This event handler is implemented.
status	This event handler is unimplemented.
previousSibling	This event handler is unimplemented.

XMLSocket ActionScript Class

This class implements an XMLSocket object.

The Methods of the Class

close() — close()
Close the socket connection.

connect() — connect()
Connect to a host over a network connection.

send() — send()
Send a message through a network connection.

The Event Handlers of the XMLSocket Class

onClose	onClose Called when a network connection is closed.
onConnect	onConnect Called when a network connect has connected.
onData	onData Called then there is a message from the network.
onXML	onXML Called when an XML message is received.

XMLSocket Class Conformance

Class Name	**Conformance**
close()	This method is implemented.
connect()	This method is implemented.
send()	This method is implemented.
onClose	This event handler is unimplemented.
onConnect	This event handler is implemented.
onData	This event handler is implemented.
onXML	This event handler is implemented.

Flash Opcodes

There are many opcodes in Flash, and Gnash implements the majority of them up to version 7 of the Flash format. Gnash will print an "unimplemented" message whenever it sees an opcode that isn't implemented. If you try to play a movie and it doesn't appear to be executing properly, run *gnash* or *gprocessor* with the *-v* option to see the debug messages. You can also use the *-w* option to *gnash* to write the debug messages to disk.

Unimplemented Opcodes

As of March, 2006, these are the few opcodes that haven't been implemented for full SWF version 7 compliance. SWF version 8 adds a few more that currently aren't listed here.

0x2A Throw	0x2A Throw Throw an error that can be caught by a *catch* statement.
0x2C Implements	0x2C Implements Specifies that a subclass must define all the derived methods.
0x69 Extends	0x69 Extends

	Define a subclass of a class.
0x55 enum_object	0x55 enum_object Push the name of each member of an enum on the stack.
0x8F Try	0x8F Try Protect a block of code during which an error may occur.
0x53 new_method	0x53 new_method Get the name of a method.
0x31 md length	0x31 md length Get the length of a multi-byte string.
0x35 md substring	0x35 md substring Get a substring from a multi-byte string.
0x37 md chr	0x37 md chr Get a single character from a multi-byte string.
0x3A delete	0x3A delete Delete an object.
0x45 get target	0x45 get target Return the path to a sprite.

The Interpreter Engine

FIXME:

The Main Loop

FIXME:

I/O Processing

FIXME:

Handling Values

All of the main values in Gnash as used by the interpreter, are usually an *as_value* class. This is a generic object to hold data. The supported data types for an object are *BOOLEAN*, *STRING*, *NUMBER*, *OBJECT*, *C_FUNCTION*, *AS_FUNCTION*. You can retrieve the value of an *as_value* using the conversion methods. For example, *to_tu_string* returns the value as string using the Gnash small STL library. Similarly, *to_number* would return this same value as a *double*.

as_value is often used as the initializer for a property or the data for a callback. This is done so the type of the object is specified along with the data.

```
// Set the callback for a new XML object
obj->set_member("XML", as_value(xml_new));

// Set the property to the value of text
obj->set_member("nodeName", as_value(text));

// Set the property to null, but at least it exists
```

```
obj->set_member("nodeValue", as_value(""));



// Get the name of an object
name = fn.env->top(0).to_string());

// Get the value of an object
value = fn.env->top(1).to_number);
```

as_value set methods

While *as_value* allows you to use any of the supported data types when invoking the constructor (as in the prior example). This is a common way to set the data and type of a value. Often it's necessary to set the value of an object after it is created, or to change the existing value. The = operator is also supported, so it is also possible to set a value and its type this way as well. I sort of lean towards the explicit style of setting a type, so here's all the methods that explicitly set a value.

as_value::set_bool(bool)	as_value::set_bool(bool) Set the value to a boolean value.
as_value::set_int(int)	as_value::set_int(int) Set the value to an integer value.
as_value::set_double(double)	as_value::set_double(double) Set the value to a floating point double value.
as_value::set_string(const char*)	as_value::set_string(const char*) Set the value to a *const char** value.
as_value::set_tu_string(int)	as_value::set_tu_string(int) Set the value to an tu_string value. Once all the containers have been converted to using standard STL classes, this method will go away.
as_value::set_nan(int)	as_value::set_nan(int) Set the value to an NaN (Not a Number) value.
as_value::set_null()	as_value::set_null() Set the value so this is a *NULL* object.
as_value::set_undefined()	as_value::set_undefined() Set the value so this is an *undefined* object.
as_value::set_as_object_interface(as_object *)	as_value::set_as_object_interface(as_object *) Set the value to an object value.
as_value::set_as_c_function_ptr(int)	as_value::set_as_c_function_ptr(int) Set the value.

as_value::set_function_as_object(int)	as_value::set_function_as_object(int) Set the value.

as_value get methods

as_value::to_bool(bool)	as_value::to_bool(bool) Return the value as a boolean.
as_value::to_number()	as_value::to_number() Return the value as a number object.
as_value::to_string()	as_value::to_string() Return the value as a *const char**.
as_value::to_tu_string(int)	as_value::to_tu_string(int) Return the value as a tu_string value. Once all the containers have been converted to using standard STL classes, this method will go away.
as_value::is_nan()	as_value::is_nan() Return true if set to NaN (Not a Number).
as_value::is_inf()	as_value::is_inf() Returns true if the number has an infinite value.
as_value::is_finite()	as_value::is_finite() Returns true if the number has an finite value.
as_value::to_object()	as_value::to_object() Return the value as an *as_object_interface*. This is often used as the "handle" for an object within Gnash. You would use this when you need to do *set_member()* or *get_member()* operations.
as_value::to_c_function()	as_value::to_c_function() Return the value as a C function pointer.
as_value::to_as_function()	as_value::to_as_function() Return the value as an ActionScript function.

Handling Objects

FIXME:

The Environment Stack

FIXME:

Sound handling in Gnash

When a SWF-file contains audio Gnash uses its sound handlers to play it. At the moment there are two sound handlers, but it is likely that more will be made.

Sound types

Sounds can be divided into two groups: event-sounds and soundstreams. Event-sounds are contained in a single SWF frame, but the playtime can span multiple frames. Soundstreams can be (and normally are) divided between the SWF frames the soundstreams spans. This means that if a gotoframe-action jumps to a frame which contains data for a soundstream, playback of the stream can be picked up from there.

Sound parsing

When Gnash parses a SWF-file, it creates a sound handler if possible and hands over the sounds to it. Since the event-sounds are contained in one frame, the entire event-sound is retrieved at once, while a soundstream maybe not be completely retrived before the entire SWF-file has been parsed. But since the entire soundstream doesn't need to be present when playback starts, it is not necessary to wait.

Sound playback

When a sound is about to be played Gnash calls the sound handler, which then starts to play the sound and return. All the playing is done by threads (in both SDL and Gstreamer), so once started the audio and graphics are not sync'ed with each other, which means that we have to trust both the graphic backend and the audio backend to play at correct speed.

The SDL sound backend

The current SDL sound backend has replaced the original sound handler, based on SDL_mixer, which by design had some limitations, making it difficult to implement needed features such as support for soundstreams. The SDL sound backend supports both event-sounds and soundstreams, using Gnash's internal ADPCM, and optionally MP3 support, using either FFMPEG or LIBMAD. When it receives sounddata it is stored without being decoded, unless it is ADPCM, which is decoded in the parser. When playing, backend relies on a function callback for retrieving output sound, which is decoded and resampled if needed, and all sound output is mixed together. The current SDL sound backend was made since Gnash needed a working sound backend as soon as possible, and since the gstreamer backend at the time suffered from bugs and/or lack of features in gstreamer. The result was the most complete and best soundhandler so far. The advantages of the SDL soundhandler is speed, and ease of use, while its only real disadvantage is that it has to be compiled with mp3 support, which some linux distrobutions will probably not like...

The Gstreamer backend

The Gstreamer backend, though not complete, supports both soundstreams and event-sounds. When receiving sounddata it stores it compressed, unless if it's ADPCM event-sounds, which it decodes by the parser. When the playback starts, the backend setup a Gstreamer bin containing a decoder (and other things needed) and places it in a Gstreamer pipeline, which plays the audio. All the sounddata is not passed at once, but in small chuncks, and via callbacks the pipeline gets fed. The advantages of the Gstreamer backend is that it supports both kind of sounds, it avoids all the legal mp3-stuff, and it should be relativly easy to add VORBIS support. The drawbacks are that it has longer "reply delay" when starting the playback of a sound, and it suffers under some bugs in Gstreamer that are yet to be fixed.

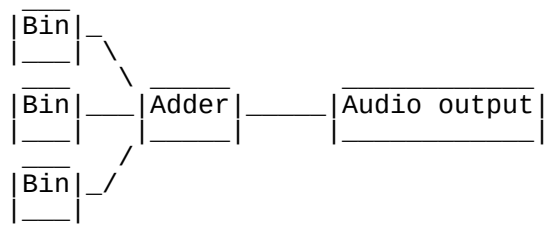
Future audio backends

It would probably be desirable to make more backends in the future, either because other and better backend systems are brought to our attention, or perhaps because an internal sound handling is better

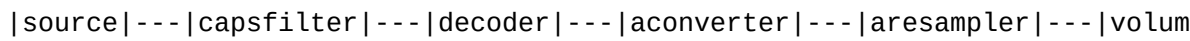
suited for embedded platform with limited software installed.

Detailed desciption of the Gstreamer backend

Gstreamer uses pipelines, bins and elements. Pipelines are the main bin, where all other bins or elements are places. Visually the audio pipeline in Gnash looks like this:



There is one bin for each sound which is being played. If a sound is played more the once at the same time, multiple bins will be made. The bins contains:



In the source element we place parts of the undecodede sounddata, and when playing the pipeline will pull the data from the element. Via callbacks it is refilled if needed. In the capsfilter the data is labeled with the format of the data. The decoder (surprise!) decodes the data. The audioconverter converts the now raw sounddata into a format accepted by the adder, all input to the adder must in the same format. The audioresampler resamples the raw sounddata into a sample accepted by the adder, all input to the adder must in the same sample rate. The volume element makes it possible to control the volume of each sound.

When a sound is done being played it emits a End-Of-Stream-signal (EOS), which is caught by an event-handler-callback, which then makes sure that the bin in question is removed from the pipeline. When a sound is told by Gnash to stop playback before it has ended playback, we do something (not yet finally implemented), which makes the bin emit an EOS, and the event-handler-callback will remove the sound from the pipeline. Unfortunantly Gstreamer has a curent bug which causes the entire pipeline to stop playing when unlinking an element from the pipeline; so far no fix is known.

Gstreamer also contains a bug concerning linking multiple elements to the adder in rapid succesion, which causes to adder to "die" and stop the playback.

Testing Support

Testing Tools

Currently Gnash uses other three tools to help with testing. Two of these are free compilers for the Flash format. This lets us write simple test cases for Gnash to test specific features, and how they need to operate.

The primary compiler used at this time is Ming [http://ming.sf.net]. Since release 0.3, *Ming* includes a command-line compiler, *makeswf*. This lets test case development happen purely with free tools.

The other tools used are optional. DejaGnu [http://www.gnu.org/software/dejagnu] is used to run multiple test cases in an automated manner. *DejaGnu* is used by many other GNU [http://www.gnu.org] projects like GCC [http://gcc.gnu.org] and Samba [http://www.samba.org].

Test Cases

ActionScript test cases are located under testsuite/actionscript.all/; these are organized as one file for ActionScript class. Other Ming-generated tests are under testsuite/ming-misc.all/; these are typically used to test specific tag types. Full movies are located in testsuite/movies.all/ and sample movies are found in testsuite/samples/. Other directories in testsuite/ are (or shall be) used for other kind of tests.

Writing ActionScript Tests

Writing ActionScript tests is very simple. The *makeswf* compiler makes use of the C preprocessor, thus allowing definition of macros and external files inclusion. We use these feature to provide common utilities for test units.

Each test unit sets an *rcsid* variable, includes the *check.as* file and performs some checks using the provided macros. Here is an example:

```
// This variable will be used by check.as
// to show testcase info as part of the test runs.
rcsid="Name and version of this testcase, usually the RCS id";

#include "check.as"

// Test object creation
check(new Object() instanceOf Object);

// Test parseInt
check(isNaN(parseInt('none')));

// Test assignment
var a = 1;
check_equals(a, 1);

// .. your tests here ...
```

The check(expr) macro will *trace* PASSED or FAILED together with the expression being evaluated and the linenumber of the check. This is the format expected by DejaGnu.

The *check_equals(obtained, expected)* macro uses equality operator == to check for equality. When possible, use of the *check_equals()* macro is preferred over *check()* in that it shows what the obtained result was in case of a mismatch. DejaGnu.

Additionally, the check.as file provides a transparent way to send results to a TextField rather then using trace. This is very useful when you happen to run a flash player without tracing support.

Test units are built by running *make TestName.swf*. This will use TestName.as as source. To build "visual" tracing version you'd run *make TestName.vswf*.

Note that if you get a syntax error from the compiler, the line number will refer to the pre-processed file. This file is called *TestName.as.pp* and it's not thrown away by *makeswf* to make debugging easier.

Sometimes an expression is only supported by a specific SWF version, or it's evaluated differently between SWF versions. For this purpose the framework provides an OUTPUT_VERSION macro that

you can use to switch code based on output version. For example:

```
#if OUTPUT_VERSION >= 7
check(_root.getSWFVersion == OUTPUT_VERSION);
#endif
```

Writing Ming-based Tests

Ming-based testcases are located in testsuite/misc-ming.all and contain a test generator and a test runner. The test generator (usually a C program) is used to produce the SWF file, while the test runner (a C++ program) will run it using a MovieTester class. Note that only the test generator needs Ming, not the test runner, so if Ming isn't installed on the user's host, the testcases can still be run as long as SWF has been distributed.

Producing tests using Ming has the advantage that you can easily see and modify the full source code for the SWF movie, and you can use some facilities provided by the Gnash testing framework to easily run tests.

For generic Ming API documentation, see http://www.libming.org [http://www.libming.org/].

Using Ming-based test generators facilities

Ming-based test generator facilities, which might be moved into a loadable SWF in the future, can be currently used by your test generator by including the ming_utils.h file and calling the appropriate functions.

The most useful facility provided for Ming-based SWF test generators is a Dejagnu-like TestState ActionScript class. In order to use this facility you must call 'add_dejagnu_functions()' right after Movie creation. The function takes an SWFMovie object and some parameters specifying depth and location of the "visual" trace textfield; it instantiates a global 'TestState' ActionScript object to keep track of test's state.

You will *not* need to directly invoke the TestState object created by the 'add_dejagnu_functions()' routine, rather you will be using other C functions which hide it's complexity:

```
check(SWFMovie mo, const char* expr, int expect_failure)

        Evaluate an ActionScript expression.
        If 'expect_failure' is true a failure is expected
        (for cases where the call exposes a known bug).

check_equals(SWFMovie mo, const char* obtained,
                          const char* expected, int expect_failure)

        Evaluate an ActionScript expression against an
        expected output.
        If 'expect_failure' is true a failure is expected
        (for cases where the call exposes a known bug).

print_tests_summary(SWFMovie mo)

        This will print a summary of tests run, and should be
        called as the last step in your SWF generator.
```

Running Ming-generated testcases

Testcases generated using Ming and the provided facilities will be self-contained, which means they can be used as tests by simply running them with whatever Player you might have. Any 'check' or 'check_equals' result will be both traced and printed in a textfield. You can use 'gprocessor -v' to have gnash use them as tests.

In order to make the test automatically run as part of 'make check', you'll need to provide a 'test runner' for your testcase and list it in the TEST_CASES variable in testsuite/misc-ming.all/Makefile.am.

A simple and generic test-runner can be found in testsuite/generic-testrunner.sh. The script can be invoked by passing it $(top_builddir) as the first argument and the name of the SWF file (without the path) as the second argument. This will create a specific runner for your test in the current build directory. A simple Makefile.am rule for doing this follows:

```
MyTest-Runner: $(srcdir)/../generic-testrunner.sh MyTest.swf
        sh $(srcdir)/../generic-testrunner.sh $(top_builddir) MyTest.swf > $@
        chmod +x $@
```

Note that there are some parts of Gnash that can NOT be tested by only using ActionScript tests. Examples include: frame advancements, actual actions execution, gui events and so on.

In this case you might want to use the MovieTester class to implement a C++ test runner. Be aware that you can *mix* tests in the MovieTester-based class with *self-contained* tests in the SWF file as long as you activate verbosity for the debuglogfile. This is done, for example, for the DefineEditTextVariableNameTest.swf file. The corresponding test runner (DefineEditTextVariableNameTest-Runner) is a C++ runner based on MovieTester class. If you run the runner you see two kinds of test results: the ones coming from the ActionScript engine, and the ones coming from the test runner. You can distinguish between the two becouse the former contains an additional timestamp and the latter does not. Also, you'll see two final summaries for the two test sets. The 'make check' rule, which uses the testsuite/simple.exp output parser as its work-horse, will count test results from both test sets.

More information about writing SWF test runners may be found in the Writing Movie testers section.

Writing Movie testers

Note

This section is a stub, must be improved.

Movie testers are executables which load an SWF, generate events (both user or system) on it, and check its state using a standard interface.

To help this process a MovieTester class is defined in the testsuite/MovieTester.{h,cpp} files; see Doxygen documentation for more information.

Note that you do NOT need access to the SWF sourcecode in order to implement a Move tester for it. Some knowledge about the expected behavior suffices.

Running The Tests

Using DejaGnu

The simple way to run the tests is to install *DejaGnu*, and use that to run the tests. That handles all the details to compile and execute the tests. To run the tests using DejaGnu, change to the *testsuite* directory and type:

```
make check
```

You can get more details by adding command line option when invoking *make*. The *make check* target in the Makefile supports a variable, *RUNTESTFLAGS* that gets passed to DejaGnu when it's invoked by *make*.

```
make check RUNTESTFLAGS="-v -a"
```

This adds the*verbose (-v)* option and the *all (-a)* option. Verbose prints much more information about how DejaGnu is running the test. It may be too much information, but if you think you are having a problem with running a test case, this is a good way to track it down. The *all* option tells DejaGnu to print all the tests that *PASS*, as well as those that *FAIL*.

Manually Running Tests

You can also run all the test cases by hand, which is useful if you want to see all the debug output from the test case. Often the debug message that come deep from within Gnash are the most useful during development.

The first step is to compile the test case. Ming's *makeswf* program is used to compile the test case into Flash. By default, no options are required. Running *makeswf* looks like this:

```
shellprompt> makeswf XML.as
Output file name: out.swf
Output compression level: 9
Output SWF version: 6
Preprocessing XML.as... done.
Compiling `XML.as.pp' into frame 1... done.
Saving output to out.swf... done.
```

Once you have the flash movie version of the test case, you can run it through the Gnash standalone player with a few options which enable a simple test cases to be run that has no associated graphics display, and consist only of unit level tests for an ActionScript class.

```
shellprompt> gprocessor -v out.swf
PASSED: XML::XML() constructor
PASSED: XML::addRequestHeader() exists
PASSED: XML::appendChild() exists
...
```

Appendix

Code Style

I know any discussion of coding styles leads to strong opinions, so I'll state simply I follow the GNU Coding Standards [http://www.gnu.org/prep/standards/standards.html]. Where there is some flexibility as to the location of braces, I prefer mine on the end of a line when using an *if*, *while*, or *do* statement. I find this more compact style easier to read and parse by eye. I'm also a big fan of always using braces around *if* statements, even if they're one liners.

Here's my tweaked style settings for *Emacs*, the one true editor to rule them all.

```
      (defconst my-style
          '((c-tab-always-indent   . t)
           (c-auto-newline    . t)
           (c-hanging-braces-alist . (
                                      (brace-list-intro)
                                      (namespace-open)
                                      (inline-open)
                                      (block-open)
                                      (brace-list-open)
                                      (brace-list-close)
                                      (brace-entry-open)
                                      (brace-else-brace)
                                      (brace-elseif-brace)
                                      (class-open after)
                                      (class-close)
                                      (defun-open after)
                                      (defun-close)
                                      (extern-lang-open)
                                      (inexpr-class-open)
                                      (statement-open)
                                      (substatement-open)
                                      (inexpr-class-close)))
           (c-hanging-colons-alist . ((member-init-intro before)
                                      (inher-intro)
                                      (case-label after)
                                      (label after)
                                      (access-label after)))
           (c-offsets-alist . (
                                      (innamespace . 0)
                                      (case-label  . 2)
                                      ))
           (c-cleanup-list . (
                                      (scope-operator)
                                      (empty-defun-braces)
                                      (brace-else-brace)
                                      (brace-elseif-brace)
                                      (defun-close-semi)
                                      (list-close-comma)
                                      )
                                )
    ;; no automatic newlines after ';' if following line non-blank or inside
    ;; one-line inline methods
    (add-to-list 'c-hanging-semi&comma-criteria
                 'c-semi&comma-no-newlines-before-nonblanks)
    (add-to-list 'c-hanging-semi&comma-criteria
                 'c-semi&comma-no-newlines-for-oneline-inliners)
;    (knr-argdecl-intro . -)
    (c-echo-syntactic-information-p . t)
    )
  "My GNU Programming Style")
```

Another coding consideration: comments are good! Over commenting isn't good. Here is an overcom-

mented example:

```
counter++;         // increment counter
```

Gnash also uses Doxygen [http://www.doxygen.org]. style comments. These are processed by Doxygen when building a cross reference of all the classes, and is a good way to help push internals documentation from the depths of the code into documentation where it can be seen by others.

Doxygen style comments for *C++* code involves simply using three slashes */// *instead of the standard two slashes *//* used for C++ comments. Here's a short comment block for the *XML::cloneNode()* method:

```
/// \brief copy a node
///
/// Method; constructs and returns a new XML node of the same type,
/// name, value, and attributes as the specified XML object. If deep
/// is set to true, all child nodes are recursively cloned, resulting
/// in an exact copy of the original object's document tree.
XMLNode &
XML::cloneNode(XMLNode &newnode, bool deep) {
...
}
```

The *\brief* keyword means that the text becomes associated when listing all the classes on the generated web pages. The text after the blank link becomes the detailed description which appears on the generated web page for that class and method.

RTMP Protocol

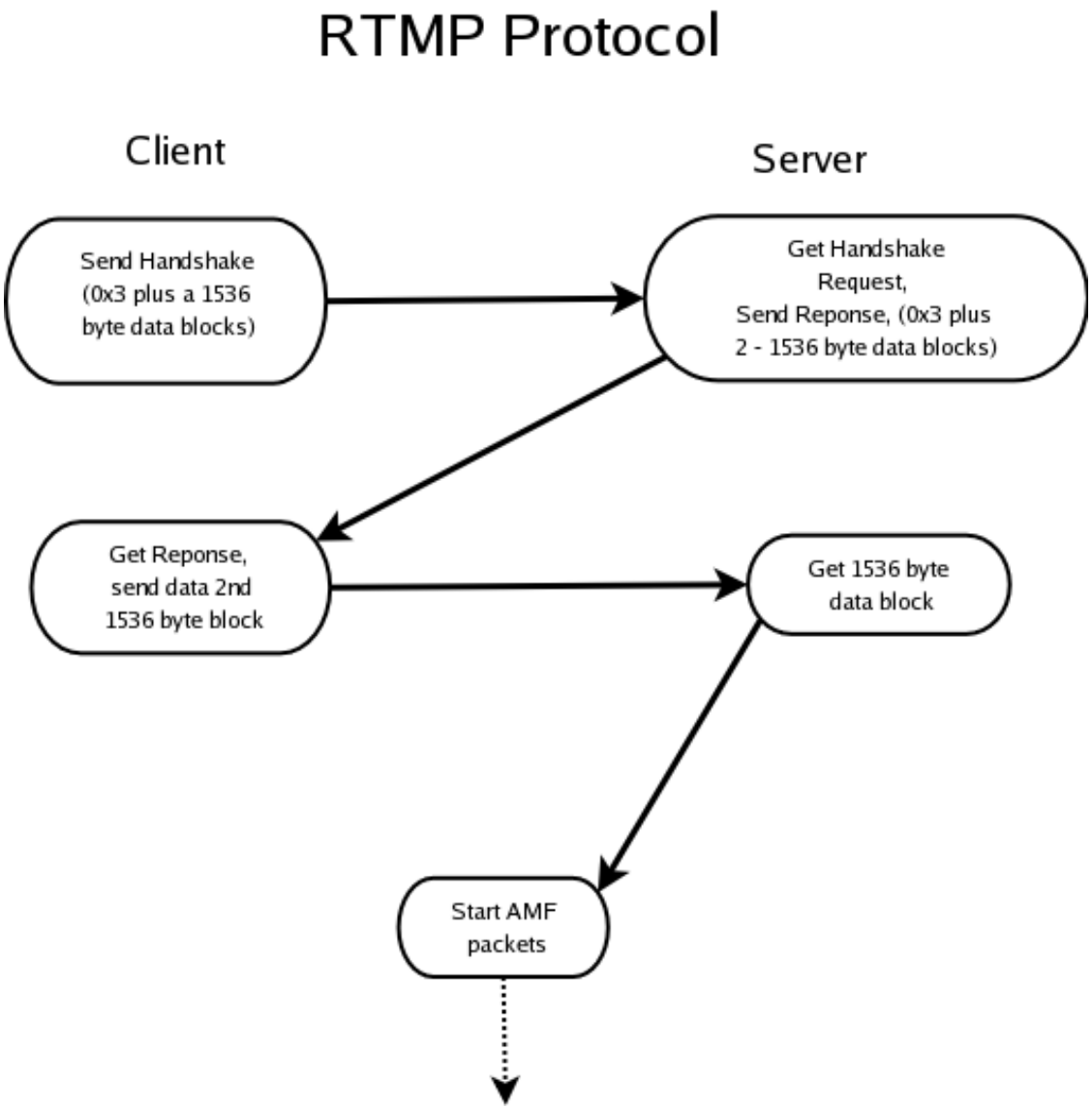
This document is based mostly on my own reverse engineering of the RTMP protocol and AMF format. *tcpdump* and *ethereal* are your friend. Some additional info that got me started was from the Red5 [http://www.osflash.org/red5] project. *Red5* is the only other open source Flash server. So some details are still vague, but as the implementation appears to work, we'll figure out what they are later.

The Real Time Messaging Protocol was created by MacroMedia (now Adobe) for delivering Flash objects and video over a network connection. Currently the only servers which support this format are the MacroMedia Media sever, and the Open Source Red5 project.

This is a simple protocol, optimized for poor bandwidth connections. It can support up to 64 concurrent streams over the same network connection. Part of each AMF packet header contains the index number of the stream. A single RTMP message can contain multiple AMF packets.

An RTMP connection uses Tcp/ip port 1935. It is also possible to tunnel RTMP over an HTTP connection using port 80. Each AMF packet is 128 bytes long except for streaming audio, which has 64 byte packets.

The basics of the RTMP protocol are as follows. All communications are initiated by the client.



The client starts the RTMP connection by sending a single byte with a value of 0x3. This byte is followed by a data block of 1536 bytes. The format if this data block is unknown, but it appears to not be actually used by the protocol except as a handshake.

The server receives this packet, stores the 1536 byte data block, and then send a single byte with the value of 0x3, followed by two 1536 data blocks. The second data block is the full contents of the original data block as sent by the client.

The client receives the 1536 byte data block, and if they match, the connection is established. After the handshake process is done, there are three other messages that the client sends to the sever to start the data flowing.

The first AMF packet sent to the server contains the *connect* packet. This doesn't appear to do much but notify the server the client is happy with the handshake, and ready to start reading packets.

The second packet is the *NetConnection* object from the client. This ActionScript class is used by the Flash movie to create the network connection to the server.

The third packet is the *NetStream* object from the client. This is the ActionScript class used to specify

the file to be streamed by the server.

The RTMP packet for our example looks like this:

```
030000190000c91400000000020007connect00?f0000000000000030003app0200#
software/gnash/tests/1153948634.flv0008flashVer02000cLNX 6,0,82,0 0006
swfUrl02001dfile:///file|%2Ftmp%2Fout.swfc30005tcUrl\002\0004
rtmp://localhost/software/gnash/tests/1153948634.flv\000\000\t
\002\000\005userx
```

We'll take this apart in a bit, but you can see how all three AMF packets are in the same message. The message is recieved in several 128 byte blocks, with the last one being less than that. The total size of the RTMP message is in the header, so the reader can tell if the entire message was read or not.

The RTMP header is first, followed by the connect message as an ASCII string as the message body. The following AMF packet is the *NetConnection* one, which specifies that this is coming from a Flash application. This also contains the file path the server can use to find the file to stream. This is then followed by the version number, which I assume is the version of the Flash player, so the server knows what it is talking to.

The third packet is the one from *NetStream*, which specifies the URL used for the movie, followed by the user name for a semblance of security.

For the next level of detail, we'll explain the format of AMF. AMF is used by the RTMP protocol to transfer data. Each Flash object is encapsulated in an AMF packet, including streaming audio or video.

The first byte of the RTMP header determines two things about the rest of the message. The first 2 bits of this byte signify the total size of the RTMP header. The RTMP header is of a variable size, so this is important.

00 00
: This specifies the header contains 12 bytes, including this one.

01 01
: This specifies the header contains 8 bytes, including this one.

02 02
: This specifies the header contains 4 bytes, including this one.

03 03
: This specifies the header contains 1 byte, so this is the complete header.

The other 6 bits in this byte represent the AMF index. As a single RTMP connection can support multiple data streams, this signifies which stream this packet is for. Once an AMF object is fully received by the client, the AMF index may be reused.

For messages with headers of at least 4 bytes, the next 3 bytes are used by audio and video data packets, but at this time the meaning of this field is unknown.

For messages with a 8 byte or larger header, the next 3 bytes determine the size of the RTMP message being transmitted. Messages with a 1 byte or 4 byte header use a standard size, 128 bytes for video, and 64 bytes for audio.

For messages with an 8 byte or larger header, the next byte is the type of the AMF object.

0x3 0x3

This specifies the content type of the RTMP packet is the number of bytes read. This is used to start the RTMP connection.

0x4 0x4
This specifies the content type of the RTMP message is a *ping* packet.

0x5 0x5
This specifies the content type of the RTMP message is server response of some type.

0x6 0x6
This specifies the content type of the RTMP packet is client request of some type.

0x8 0x8
This specifies the content type of the RTMP packet is an audio message.

0x9 0x9
This specifies the content type of the RTMP message is a video packet.

0x12 0x12
This specifies the content type of the RTMP message is notify.

0x13 0x13
This specifies the content type of the RTMP message is shared object.

0x14 0x14
This specifies the content type of the RTMP message is remote procedure call. This invokes the method of a Flash class remotely.

There are two sets of data types to consider. One set is used by the to specify the content type of the AMF object, the other is an ActionScript data type tag used to denote which type of object is being transferred.

The values of the initial type byte are:

0x0 0x0
This specifies the data in the AMF packet is a numeric value. All numeric values in Flash are 64 bit, *big-endian*.

0x1 0x1
This specifies the data in the AMF packet is a boolean value.

0x2 0x2
This specifies the data in the AMF packet is an *ASCII* string.

0x3 0x3
This specifies the data in the AMF packet is a Flash object. The Flash object data type field further along in the message specifies which type of ActionScript object it is.

0x4 0x4
This specifies the data in the AMF packet is a Flash movie, ie. another Flash movie.

0x5 0x5
This specifies the data in the AMF packet is a NULL value. NULL is often used as the return code from calling Flash functions.

0x6 0x6
This specifies the data in the AMF packet is a undefined. This is also used as the return code

from calling Flash functions.

0x7 0x7
This specifies the data in the AMF packet is a reference.

0x8 0x8
This specifies the data in the AMF packet is a ECMA array.

0x9 0x9
This specifies the data in the AMF packet is the end of an object definition. As an object is transmitted with multiple AMF packets, this lets the server know when the end of the object is reached.

0xa 0xa
This specifies the data in the AMF packet is a Strict array.

0xb 0xb
This specifies the data in the AMF packet is a date.

0xc 0xc
This specifies the data in the AMF packet is a multibyte string. Multibyte strings are used for international language support to represent non *ASCII* fonts.

0xd 0xd
This specifies the data in the AMF packet is a an unsupported feature.

0xe 0xe
This specifies the data in the AMF packet is a record set.

0xf 0xf
This specifies the data in the AMF packet is a AML object. XML objects are then parsed by the *XML* ActionScript class.

0x10 0x10
This specifies the data in the AMF packet is a typed object.

For messages with a 12 byte header, the last 4 bytes are the routing of the message. If the destination is the server, this value is the NetStream object source. If the destination is the cllient, this is the NetStream object for this RTMP message. A value of 0x00000000 appears to be reserved for the NetConnection object.

Multiple AMF streams can be contained in a single RTMP messages, so it's important to check the index of each AMF packet.

An example RTMP header might look like this. (spaces added between fields for clarity) All the numbers are in hex.

```
03 000019 0000c9 14 000000000
```

03 03
The first two bits of this byte are the size of the header, which in this example is 00, for a 12 byte header. The next 6 bits is the AMF stream index number, which in this example is 0x3.

000019	000019 These 3 bytes currently have an unknown purpose.
000c9	000c9 Since this example has a 12 byte header, this is the size of the RTMP message, in this case 201 bytes.
14	14 This is the content type of the RTMP message, which in this case is to invoke a remote function call. (which we later see is the connect function).
00000000	00000000 The source is the NetConnection object used to start this connection.

AMF Format

The AMF format is used in the LocalConnection, SharedObject, NetConnection, and NetStream ActionScript classes. This is a means of binary data interchange between Flash movies, or between a Flash player and a Flash server.

Like the RTMP messages, an AMF packet header can be of a variable size. The size is either the same as the initial header of the RTMP message, or a 1 byte header, which is commonly used for streaming audio or video data.

The body of an AMF packet may look something like this example. The spaces have been added for clarity.

```
02  0007 636f6e6e656374
```

02	02 This is a single byte header. The value of the first 2 bits is 0x3, and the AMF index is also 0x3.
0007	0007 This is the length in bytes of the string.
63 6f 6e 6e 65 63 74	63 6f 6e 6e 65 63 74 This is the string. Note that there is no null terminator since the length is specified.

Authors

Gnash is maintained by Rob Savoye <rob@senecass.com>. Other active developers are: Sandro Santilli, Bastiaan Jacques, Vitaly Alexeev, and Tomas Groth. Please send all comments and suggestions to <gnash@gnu.org>. Please use this link to submit a patch [https://savannah.gnu.org/patch/?group=gnash] or this link to file a bug report [https://savannah.gnu.org/bugs/?group=gnash]. I am available for consulting on a variety of renewable energy and open source technologies. More details at http://www.senecass.com

The primary author of GameSWF (which provided the initial code for Gnash) is Thatcher Ulrich <tu@tulrich.com>. Other individuals who contributed code are: Mike Shaver, Thierry Berger-Perrin, Ignacio Castaño, Willem Kokke, Vitaly Alexeev, Alexander Streit, and Rob Savoye.

A. GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- **A.** Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that ver-

sion gives permission.

- **B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
- **C.** State on the Title Page the name of the publisher of the Modified Version, as the publisher.
- **D.** Preserve all the copyright notices of the Document.
- **E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- **F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- **G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- **H.** Include an unaltered copy of this License.
- **I.** Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- **J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- **K.** In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- **L.** Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- **M.** Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
- **N.** Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of

Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version .

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and dispbibute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document , on account of their being thus compiled, if they are not themselves derivative works of the Document. If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission

from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation [http://www.gnu.org/fsf/fsf.html] may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/ [http://www.gnu.org/copyleft].

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

Addendum

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License [http://www.gnu.org/copyleft/gpl.html], to permit their use in free software.