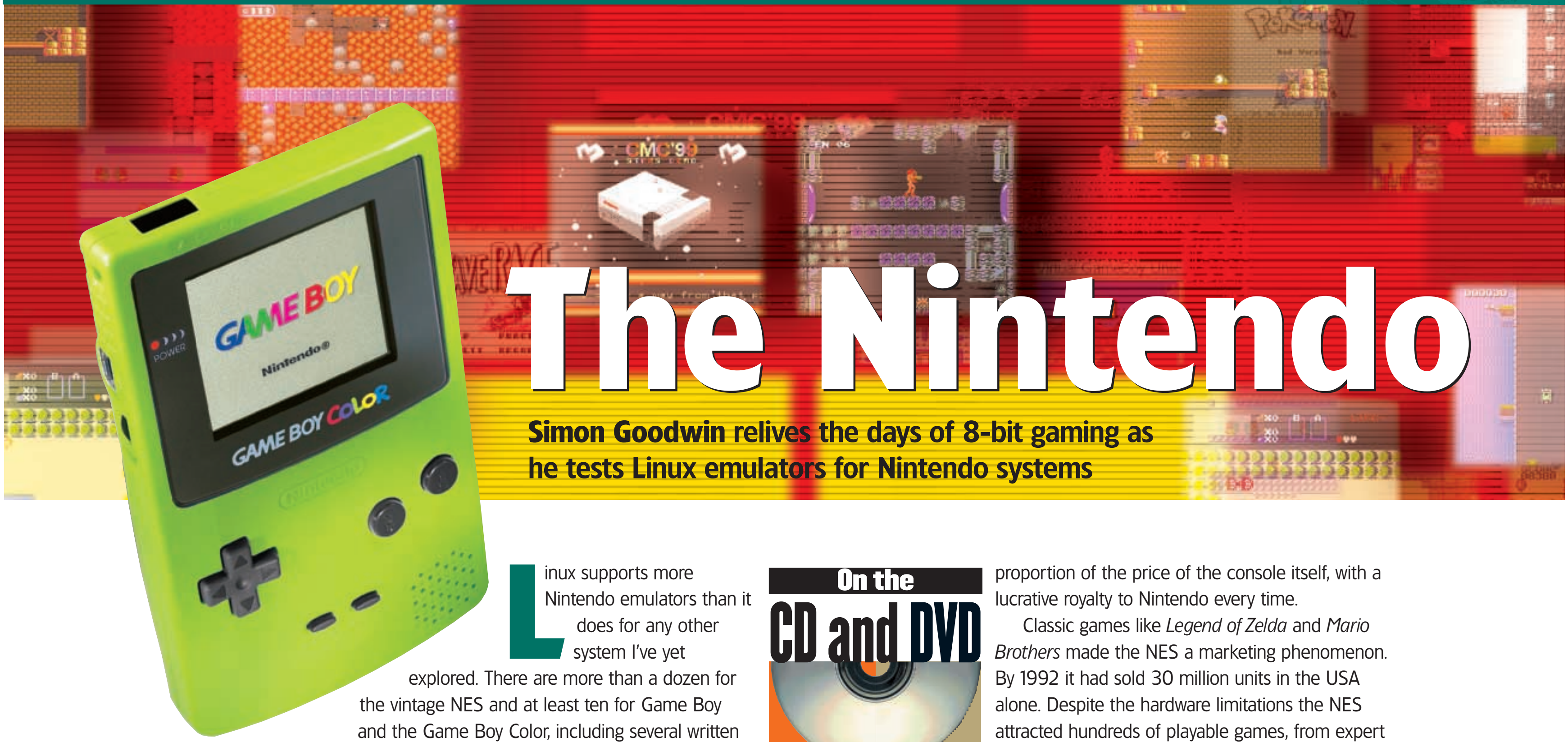


The Nintendo

Simon Goodwin relives the days of 8-bit gaming as he tests Linux emulators for Nintendo systems

On the CD and DVD

Linux supports more Nintendo emulators than it does for any other system I've yet explored. There are more than a dozen for the vintage NES and at least ten for Game Boy and the Game Boy Color, including several written in Java. There are also Linux emulators for the Super Nintendo, Reality Boy and Game Boy Advance systems, which we'll test another time. Non-commercial software for these systems is abundant, so there's no need to copy ROMs to gain a taste for Nintendo emulation.

Legend Of Zelda

The Nintendo Entertainment System, or NES, was the first successful games console from the Far East, where it was branded Famicom. It capitalised on the slump in video game sales in the States in the mid-1980s. Nintendo stepped into the gap, and have since sold more than 100 million systems worldwide. That's not bad for a firm that began the last century manufacturing playing cards. In the early '80s Nintendo moved into video games, programming classics such as *Donkey Kong* for Atari coin-op hardware. The NES is little more advanced than the Atari consoles it usurped. It uses a 6502 processor like early Apple and Commodore micros, and a simple tile-based graphics system which built up screens from a grid of small squares.

This approach enabled Nintendo to keep the console simple and pack lots of graphics and gameplay into small ROM (Read Only Memory) cartridges. Each title sold for an appreciable proportion of the price of the console itself, with a lucrative royalty to Nintendo every time.

Classic games like *Legend of Zelda* and *Mario Brothers* made the NES a marketing phenomenon. By 1992 it had sold 30 million units in the USA alone. Despite the hardware limitations the NES attracted hundreds of playable games, from expert arcade game developers like Namco (*Pacman*, *Galaxian*, *DigDug*), Konami (*Castlevania*, *Ninja Turtles*), Taito (*Space Invaders, Arcanoid*) and Nintendo themselves. Add-ons such as Action Replay and Game Genie allow cartridges to be customised, boosting play value and leading enthusiasts to explore the innards of Nintendo systems. Now they can use their favourite cheats, and more, with Nintendo emulators.

Tile Patterns

Nintendo taxed developers heavily for the privilege of writing for the NES, vetted their software and tried to keep details secret. This had the desired result of keeping software prices — and, in general, quality — high, but makes emulation challenging.

The NES console contains 10K of RAM, mostly allocated to graphics. It has two digital joystick ports and a TV output. Five mono audio channels offer PCM for short samples, 'noise', and three periodic pulse or triangle waveforms.

The display background is built up from tiles eight pixels square. 16 colours are represented by four bits. Two bits are stored with each tile, and the other two come from attributes which select four from 16 colours, in 32x32 pixel blocks. The background image in memory is 256 pixels square, trimmed

GameBoy emulator comparison table

EMU,	VERSION	SOUND	COLOR	CHEATS	GUI	STICK	RESIZE	MONITOR	LINK	SOURCE	DISPLAY
Cingb,	0.26A	✔	✔	✔	✘	✔	✔	✘	✔	✔	X/FB
GameBoyEmu	9/3/01	✘	✔	✘	✔	✘	✔	✔	✘	✔	Java
GBe	0.2.22	✘	✘	✘	✔	✔	✔	✘	✔	✔	X/VGA
GnuBoy,	1.0.1	✔	✔	✘	✔	✔	✔	✘	✘	✔	X/FB/SDL/VGA
JavaBoy,	0.8	✔	✔	✘	✔	✘	✔	✔	✔	✔	Java 1.2+
PGB,	0.87	✘	✔	✘	✔	✘	✔	✔	✘	✔	Java
VengaBoy,	1.01	✘	✔	✘	✔	✔	✔	✘	✘	✔	X
VGB,	1.6/2.1	✔	✔	✔	✔	✘	✔	✔	✔	✘	X

"If these Linux-friendly GameBoy emulators are not enough, seek out a stable version of JGB or Boycott."
"KEY: FB = Video Frame Buffer, X=X11 window, VGA=SVGAlib screen"

Tendency

Nintendo connections

DarcNES: http://www.dridus.com/~nyef
FCE Ultra: http://fceultra.sourceforge.net
FCE SDL: http://digicharat.8m.com
GB Color demos: http://www.subport.org/files/demos
GameBoy FAQs: http://www.dmgice.com/hardware
iNES and VGB: http://www.komkon.org/fms/comp
InfoNES: http://www.geocities.co.jp/SiliconValley/5604/infones.html
Java GBemu: http://www.btinternet.com/~davidwinchurch
JavaBoy: http://www.millstone.demon.co.uk
JGB: http://www.palevich.com/JavaGameboy
NES demos: http://www.zophar.net/roms/files/nes
NES coding: http://devlib.virtualave.net/dir/files/nes
PGB: http://www.sev.org/users/bmazur/
TuxNES: http://tuxnes.sourceforge.net
Various: http://linuxberg.tele.net/files/x11/entertain
Various: http://www.zophar.net/unix/Files
VGB GUI: http://www.daimi.au.dk/~maxx

• Our cover CD contains dozens of freely-distributable emulators plus freebie games and demos. Zophar and Linuxberg have big Nintendo emulator archives, though these are not always complete or the latest files. You'll also find collaborative developments on SourceForge, and *GnuBoy*, *iNES* and *NEStra* packages at Debian.org.

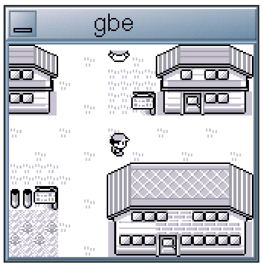
vertically to 240 pixels for European displays and 224 for the US and Japan. Thus a screen can be encoded in just 1024 bytes for the tile numbers, and 64 attribute bytes. Each tile pattern needs 16 bytes, but tiles can be re-used and re-coloured. Large scale animation is possible by changing patterns during vertical blanking of the display. A big map can be packed into a small ROM. Typical NES games use repetitive backgrounds with player characters and targets superimposed. Sprites are overlaid and stored similarly, except sprite tiles can be twice as tall and have a separate palette. Up to eight sprites can appear on each line, and they can re-use background patterns to save memory.

Mapper Numbers

Scrolling and panning depend partly on the cartridge, rather than just the console hardware. Emulator switches toggle 'vertical and horizontal mirroring' if the default gives a scrambled screen; other options support scan-line changes needed for some games.

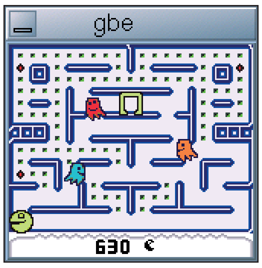
Cartridges may contain code or graphics in ROM, video memory, non-volatile RAM for saved games, and invariably a 'mapper' chip to funnel their contents into the 16-bit address space of the 6502. The original limit of 32K bytes per program — generously described as 256K (in bits!) by Nintendo — was soon a restriction, and so the hardware was extended to allow more memory, by 'mapping' it piecemeal into the eight-bit processor's space. Emulators must mimic not only the console but also the mapper in each cartridge.

Low mapper numbers are the most common. 0 is the simplest, a single-page ROM. *Zelda* uses Mapper one to control its 128K. Konami games use mapper two, which gives one fixed and one switchable 16K page. Mapper three uses a fixed program and switchable graphics pages, mapper four allows scan-line interrupts, and so on. Such variations are one reason why there are so many Linux NES emulators — what works best for some cartridges may make emulating others problematic.



I know I left a pocket-full of monsters somewhere...

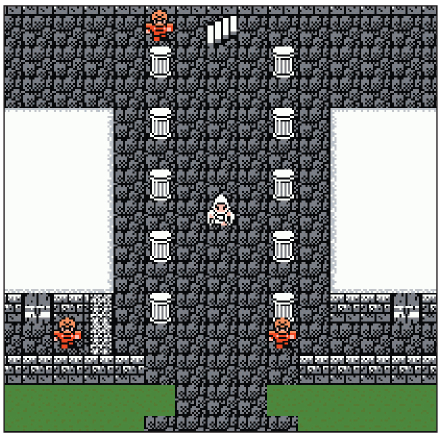
It's a pity this *R-Type* screen is not animated.



***Pokemon Red* is strangely grey on a Game Boy.**

GAME BOY

In 1989 Nintendo followed up their NES with a tiny pocket games machine, similar to their one-game LCD (Liquid Crystal Display) units of a decade earlier, but designed to run the lucrative plug-in cartridges. This product, the Game Boy, proved playability and long battery life was more important than graphical sophistication in the hand-held market. So far it has sold more than 60 million units, and attracted the release of more than a thousand commercial cartridges. The Game Boy has an eight-bit processor which resembles the Intel 8080 and Zilog Z80, with extensions and omissions. It has 16K of RAM and supports up to 512K »



The first outing for the *Final Fantasy* franchise on NES — it's come a long way since.

« ROM cartridges. The 160 by 144 pixel screen is small and monochrome, but most emulators can resize it to make better use of a desktop display, and at least the LCD ghosting is eliminated in emulation on a CRT. The Game Boy has four simple sound channels, split in stereo.

The longevity of the Game Boy baffled benchmarks and pixel-counters, who saw faster and far more colourful systems like the Atari Lynx and Sega GameGear come and go while Game Boy refused to die. This prompted Nintendo to produce a colour version in 1998. Many of the Linux Game Boy emulators have been updated to run Color (sic) software. The Game Boy Color has much the same processor but clocked at close to 8 MHz. Emulation is more demanding but within the capability of most desktop computer processors made in the last decade.

GnuBoy

GnuBoy is a much-parted Game Boy emulator with Color support. The code should compile on all POSIX systems but can use special i386 optimisations. In X sound and animation were good at the default tiny size and scale factors 2 and 3, but could not keep up at the maximum scale 4, giving choppy results. I was happy to forgo the custom screens of fb.device, SVGAlib and SDL as **xgnuboy —scale=3** gave smooth response and an acceptably large X window. fbgnuboy ran *Bubble Bobble* acceptably, though confined to a small square in the middle of the frame buffer, with slight hiccups in the audio and graphics, but I found its —scale switch rendered everything bigger but in black on very dark red, whatever the buffer depth or size.

Virtual Game Boy

Marat's *VGB* targets Game Boy programmers. *VGB* is controlled with command switches and function keys, and supports multi-player gaming over TCP/IP. It was much too fast on my machine till I tamed it with:

```
vgb -sound 48000 -scale 2 -sync 75 -uperiod 1
```

This gives a quadruple-sized display synchronised to each frame of my 75 Hertz monitor, with 48KHz sound. The screen size can be scaled continuously by dragging the corner of the X window, and this is another way to adjust speed. Version 2.1 sounds better but seems less compatible — *Bubble Bobble* bombed back to the menu with an 'unrecognised instruction' report after incorrectly drawing the first screen, but version 1.6 worked fine, though more buzzily. VGB's Color extensions worked well with all the ROMs I tried. The *Phoenix* demo has a bad ROM checksum, but the switches -gbc -nocrc allow VGB to run it.

CINGB and GBe

Cinek's Game Boy emulator has basic sound, a switch to double the X window size, and an optional debugger with video snooping but no disassembler. It's quite good in X, but I couldn't get the option to directly write and read a frame buffer more than 8-bits deep to work. *GBe* is small and simple but wants for sound, documentation and testing. Compatibility is patchy. It compiled once I added libreadline4-dev from the third Debian CD to my installation. It can run on your X desktop or in a custom SVGA screen, albeit silently. The text file trails off after a few lines of usage information, noting that the author is 'just too lazy to finish this documentation.' You'll probably need to know that the

NES Natives

The table summarises mainstream NES emulators with common features. darcNES started out NES-specific, but now emulates other consoles too. fwNES is another SVGA emulator. Little John has a neat GUI, and uses SDL 1.0. while NoFrendo requires the later SDL 1.2.

FCE-Ultra

FCE-Ultra follows the obsolete xFCE, and NextFCE for BSD Unix, derived from the MSDOS *FamiCom* Emulator. You must be root to launch it, and pick the right video mode. SVGAlib garbles the display unless its default matches your command line; vmodes 4 and 5 suited my VESA setup. More portable SDL versions are available, based on older source. *FCE-Ultra* supports FDS disk and VS Unisystem coin-op expansion as well as basic NES hardware. It supports network play, joysticks and even emulates a *Zapper* lightgun using the mouse. It ran *Elite* without glitches in full-screen VGA window when I specified -pal -vmode 4. Without -pal it shows clipping, like most Java emulators, pointing to a PAL/NTSC issue.

InfoNES

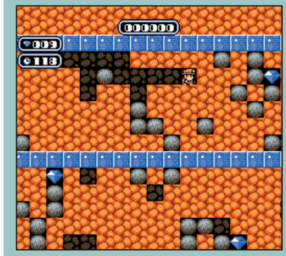
InfoNES comes compiled for RedHat 6.1 running Linux 2.2 with Xfree86 3.3 and GTK 1.2, or intriguingly Sony PlayStation 2 Linux. It puts up a file requester when launched, but lacks menus. Keys can reset or quit but there's no way to select another ROM without restarting. Alas, most Linux NES emulators share this fault. The brief HTML documentation says that *InfoNES* needs a 350 MHz Pentium 2 processor. Mapper support and compatibility with cartridge technical tricks is good, but sound became choppy once my 500 MHz K6 got busy — even window operations caused audible clicks. InfoNES ran everything I tried, including Ian Bell's demanding NES port of *Elite*, without the juddering screen of most NES emulators.

iNES

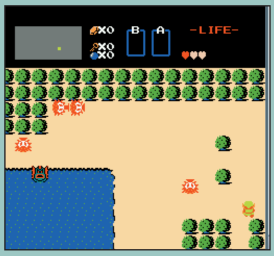
iNES is from Marat Fayzulin. Like his VGB, it's portable but not open source. Pre-compiled binaries are available for many flavours of Unix, but Mac and Windoze versions are commercial so you can't compile your own. I tested a Debian x86 package on an AMD K6/2 500, adding the zlib1 package for compressed cartridge image loading. *iNES* ran too fast without the command line option -sync 60 to slow frame rate. It's claimed to run OK on systems with a 90 MHz Pentium or equivalent. By default one frame in three is rendered to the X window. When I turned this up with -uperiod 1 it reduced the my system to a crawl. Sound was choppy, whatever configuration I tried, and *iNES* locked up several times, ignoring keys and caning the CPU. The binary-only distribution means you're unlikely to get good results unless your system closely matches Marat's own. This is a pity because *iNES* is rich in features, even emulating Famicom disk systems.

NEStra and TuxNES

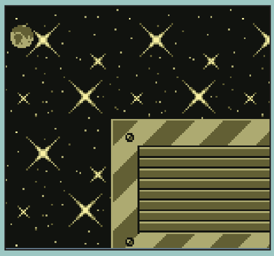
NEStra is a dynamic compiling emulator. The CPU core is impressive but the emulator is short of features, though it works. *TuxNES* is an open-source spin-off that can render via X or GGI, but also only works on x86-based



***Boulderdash* plays to the strengths of NES.**



***Zelda* diverted millions of heads from TV to NES.**



***VGB's* Space demo is neat but doesn't do much.**

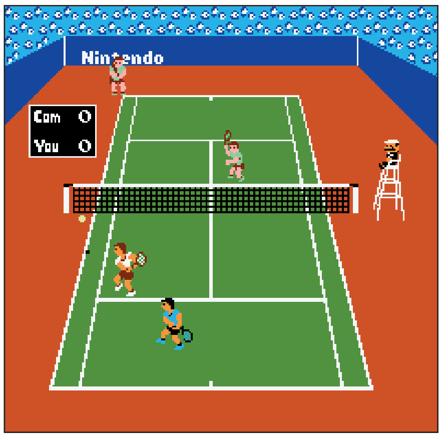
Does anyone remember *Wave Race* on the NES?

This unlicensed *Mario* clone is free to download.

control keys are Alt, Ctrl, Enter and arrows, and that it only runs in a 16-bit X desktop. Esc quits the emulation, saving game status in ~/.gbesav/. Control C calls up a simple Z80 debugger in the console from which the emulator was launched. Enter 'h' for a list of the commands in that mode, and 'e' to return.

OLD BOYS

I failed to get *Boycott* and *VengaBoy* to work despite email to the authors, now busy emulating the Game Boy Advance and NeoGeo. Old versions are available but developmentation has ceased. *VengaBoyTK* is a spin-off with a Tcl/Tk front end.



You can have hours of fun playing spot the ball with Nintendo's tennis cartridge.

Swordsman Kage whittles ninjas in the tree-lined world of *Kage* on NEStra.

Java Boys

As with the NES, portable Game Boy emulators have been written in Java, and run as well on Linux as anywhere else, though tardily compared with native emulators. *JGB* is a hybrid of the Spectrum emulator *Jasper* and *VGB 0.7*; the author concedes it's 'very slow and not very compatible' but he's looking for co-developers. *JavaBoy*, *PGB* and *GameBoyEmu* are all relatively capable and source is freely available.

JavaBoy has a good debugger and copious menus offering goodies like redefinable keys, four screen sizes and five skip rates. It even does a good job of recolouring original Game Boy software. Such games ran well, with undistorted sound, as long as I kept the screen size and update rate down. At best it could manage 30 frames per second without scaling. Game Boy Color software struggled to manage ten frames per second on my system, with intermittent sound and odd graphics blocks showing flaws of the display emulation. But *JavaBoy* is well suited for classic GameBoy ROMs, given a fast machine and JVM. Pgb is slightly faster but lacks sound and had some sprite-handling problems. I also found that the set keys option sometimes caused the emulator to lock up. *GameBoyEmu* is available as an applet or source that you can compile. It was the only Java GameBoy that gave enough space to show the key configuration clearly, and has half a dozen optional debug screens. The screen size can be standard or doubled in X and Y, not three or four times normal, but *GameBoyEmu* has unusual, though slow, image blur and edge enhancement options, and an interactive palette editor.

I compiled it on the current Sun SDK without problems bar warnings that calls to readline in the applet version were deprecated. The emulator started up, but attempts to load any ROM gave a null pointer exception. *GameBoyEmu* appears neatly written, with some unusual extras, but it lacks sound and needs more testing and documentation. LXF

systems, running Linux or BSD Unix. It's supplied as source and configured and compiled easily. The optional gtuxnes startup GUI looks neat, thanks to GTK, but is not terribly useful once you've worked out your favourite configuration as it cannot change settings on the fly.

xNES and RockNES

xNES reached version 0.5 but was abandoned in 1998. I got it sort-of working in VGA and VESA modes, albeit with eight game images in a strip across the top of the screen. SVGA is a moving target, and I suspect that you'd need a vintage video card to run xNES now. RockNES builds upon the ashes of xNES and is developed with Debian and Allegro 3.9. *RockNES* emulates the Famicom computer FDS disk system as well as the cartridge-based console. A P200 or better is strongly recommended for this strict emulation, with scan-line graphics, accurate NES sound and almost 100 mappers. Like most NES emulators, *RockNES* can load and save state information, and also PCX screen snapshots, battery-backed data files, and gameplay sequences, called 'movies', under function-key control. Everything goes into subdirectories of ~/.rocknes, so you need links to share files with rivals.

NES in Java

Jamicom was the first Java NES emulator but has been abandoned, and I was not able to get it to work. The latest archive includes JDK 1.3 and 'general' code, but no source and little documentation. *NEScafe* (groan!) is better. It uses the Java Sound API to deliver sound as well as graphics, includes a simple debugger and a server to fetch games, and is now open source. Five menus with 22 options allow toggling of sound channels, WAV file recording, and screen snapshots in TGA format. You can save and restore the machine state in <rom>.nss files which is useful when attempting tricky parts of a game. The top row of keys let you skip from 0 to 7 frames between updates to bolster speed, and tweak the tint and hue of display colours. The option to load a new cartridge doesn't save the directory between selections, consistently defaulting to the one containing the emulator code (a general Java problem) and if you want to use the debugger you must say so before loading a ROM. The HTML documentation tells you requirements, controls and how to get started, but for detailed information you'll have to consult the source code.

NES emulator comparison table

EMU	VERSION	CHEATS	JOYSTICK	SOUND	GRABS	GUI	RESIZE	SOURCE	DISPLAY
darcNES	2001	✗	✓	✓	✗	✗	✗	✓	X/VGA
FCEultra	0.70	✓	✓	✓	✓	✗	✓	✓	VGA/SDL
fwNES	0.151	✗	✓	✓	✓	✗	✓	✗	VGA
iNES	1.0	✓	✗	✓	✗	✗	✗	✗	X
InfoNES	0.75J	✗	✗	✓	✗	✗	✗	✗	X
JamiCom	1.3	✗	✗	✗	✗	✗	✗	✗	Java
Javanese	1.0	✗	✗	✗	✗	✗	✗	✗	Java 1.1+
LittleJohn	0.2.1	✗	✓	✓	✗	✓	✓	✓	SDL 1.0+
NEScafe	0.41	✓	✗	✓	✓	✓	✗	✓	Java 1.3
NEStra	0.66	✗	✗	✗	✗	✗	✗	✓	X
NoFrendo	2.0	✗	✓	✓	✓	✓	✓	✓	SDL 1.2
RockNES	0.99	✗	✓	✓	✓	✗	✓	✗	Allegro
TuxNES	0.75	✓	✓	✓	✓	✓	✓	✓	X/GGI
xNES	0.5	✗	✓	✓	✗	✗	✓	✓	VGA

Fourteen emulators should be enough for anyone, surely?