Inferno 3rd Edition 3 October 2001 Update
Release Notes

Vita Nuova
support@vitanuova.com
3 October 2001

1. Contents

This set of updates applies to the 18 June 2001 base. The installed software must therefore be that of 18 June 2001, whether installed directly, or the result of updating the original July 2000 release with update packages.

This update offers the following, compared to the June 2001 release.

Repairs and changes

Various minor bug fixes and improvements appear here and there. They include corrected usage messages, status returned to the shell on error, diagnosing failures to load library modules, and use of the `%r` format to improve diagnostics. Many *wm* applications adjust their display to suit small screens (eg, on the Compaq iPAQ); currently the applications do some of the work themselves but it will soon be automatic. Other changes are listed below.

Date(1) takes the options `-u` (show UTC/GMT) and `-n` (print time as a number). If given a number of seconds as its argument, *date* takes that as the time to convert.

Format(8) supports tiny partitions.

`/appl/env.b` uses *env*(2) to access environment variables.

For packages created after Saturday 8 September 2001, `install/create` converts the leading `10` to the letter `A` so that the names of update files remain in sorted order.

`install/inst` takes a `-c` option to cause it to carry on even if some files or directories cannot be made or updated. (This option is intended for special applications and should not generally be used when applying system updates.)

`install/install` takes a `-P` option that lists the packages to install. The `-g` option causes `install` to install all available packages.

The installation software regards files that have had carriage returns inserted before newlines as identical to the original files when deciding when files have been modified locally.

The `wm` applications now check the whole string received on the window control channel returned by *wmlib*(2)'s `titlebar,` not just an initial letter; in particular, they check for `exit` not just `'e'`.

Wm(1) takes the option `-s` to suppress the start menu.

Keyring-gensk(2) and *createsignerkey*(8) provide experimental support for the RSA algorithm for signatures instead of the default El-Gamal algorithm; `/keyring/rsaalg.c` is new.

`/keyring/egalg.c` uses the correct structure for Public keys.

Listen(1) has a new option `-i` that takes a shell command for each listener to run to set up appropriate context (eg, name space) before listening for incoming calls.

man(1) allows non-numerics in section names.

`/appl/cmd/mkfile` now includes `/mkfiles/mksubdirs` and has acquired the list of `cmd` subdirectories from `/appl/mkfile`.

Puttar gives warnings not fatal errors when files or directories cannot be found.

sh-expr(1) implements the `!=` operator.

Tail(1) no longer gives an array bound error when a binary file does not end with a newline.

`tiny/sh` does not break when *filepat*(2) is not available.

Webgrab(1) has several repairs to its HTTP protocol implementation, making it work with multi-homed servers.

`wm/sendmail` no longer fails to save messages when requested(!).

Arg(2) allows re-use by ensuring that its globals are reinitialised by its `init` function.

Bufio(2) correctly implements relative seek.

Convcs(2) provides several more character sets.

Cs(8) does not complain if it is already running, suppressing a previously confusing diagnostic.

`lib/deflate` no longer fails on the output of some PC versions of *gzip*.

Wmlib(2) adapts a little better to different font and screen sizes (though more remains to be done); on small screens, defined as those less than 480 pixels wide, *wm*(1) puts all windows at the screen origin by default.

Interface changes and extensions

Wm-deb(1) has got a `stack` button to bring up the stack window if it has previously been dismissed. It also allows breakpoints to be set in modules that have not yet been loaded.

`wm/memory`'s display has been redesigned to be more informative.

`wm/tetris` allows the use of a stylus (or mouse) to guide the placement of pieces.

`wm/view` supports PNG format

`wm/view` has a new option `-i` for use in *plumbing*(6) files, to cause it to listen for messages from the *plumber*(8). Existing plumbing files that invoke `wm/view` will typically need to change to add that option for image viewing using the plumber to work as expected.

Newns(2) provides more general option parsing by using *arg*(2), and does error checking unless the `-i` option is given to the *namespace*(6) commands. Its internal `mount` commandalso accepts the new options `-k` *keyfile* to select the source of the authentication key, and `-C` *alg* to select the encryption algorithm.

Plumber(8) takes a `-v` option to cause it to log the contents of messages (for debugging plumbing applications), and also the option `-c` *wmchan* to select an alternative window manager channel instead of `/chan/wm` when the `-w` option is used.

Tk changes and extensions

Many of the Tk changes enforce rules and do more error-checking than before, but there are several extensions and interface changes as well.

Tk applications must create the parent widget before its children. Currently the check to enforce this rule has been suppressed, but it will be enabled in future.

Widget names are now checked for well-formedness: neither trailing dot nor double dot are allowed.

When text in a text widget is deleted, embedded windows in that text are only deleted if they are descendents of the text widget.

Text widget now redisplays correctly when an embedded window is destroyed.

Text widget now checks for embedded windows in the text that have been destroyed since they were added to the text.

Widgets packed under a destroyed widget that are not descendents of that widget are now removed correctly from the packing hierarchy.

`-activebackground` now changes the border appropriately, same as `-background`.

`scrollbar` now returns currently activated part when `activate` is called with no arguments.

Only one part of a scrollbar may be active at any one time.

Tk can now distinguish between a null argument `{}` and a missing argument.

Creating an embedded window in a text widget at index 1.0 no longer causes the packer to go into an infinite loop.

Changing the options on an embedded window in a text widget caused an uninitialised pointer access.

Changing the window associated with an embedded window item in text and canvas widgets previously did not disassociate the old window correctly.

Changing the window associated with an embedded window item in a text widget did not set the widget's size appropriately.

The `-baseline` alignment option for embedded windows in text widgets previously did not calculate the line height correctly.

The `-relief` setting for buttons is now restored after enter/leave or selection.

Buttons 4, 5 and 6 have been added (for the iPAQ).

Tk no longer crashes if the 'grab' changes during the processing of a mouse event.

A new event `<Destroy>` can be bound to a widget to receive notification when it is destroyed (eg, by the destruction of a parent widget). The `<Configure>` event is propagated to slaves as well as the configured master. These two changes make it easier to implement pseudo-widgets such as *dividers*(2).

`-anchor` has been implemented for labels; `-justify` should be implemented as documented.

Submenus are unmapped correctly.

Tk detects command loops (by limiting recursion depth).

`canvas` has a new boolean option `-buffered` that controls whether the whole canvas, or just the visible region, is allocated an off-screen buffer image. It defaults to just the visible area only.

`canvas` has new operations `screenx` and `screeny` to map canvas coordinates to screen coordinates.

New commands and modules

A collection of small and tiny playing card images have been added, in `/icons/smallcards` and `/icons/tinycards`.

`install/wfind` lists the versions of a given file in a set of installation packages.

Touchcal(8) provides touch-screen calibration; it runs both inside and outside the window manager *wm*(1). Both internal and external interfaces are completely different from previous versions.

Wm-keyboard(1) describes new commands `wm/keyboard` and `wm/pen` that provide soft keyboard and single-stroke gesture recognition for touch screen devices.

Gamesrv(4) provides a file system interface for multi-player networked games; *gamesrv*(2) provides the interface for the game-specific engines loaded on demand by the game server.

`utils/awk` is a new directory containing the source for a version of *awk* for use in doing Inferno ports for systems that lack it (or a sufficiently recent version), including Windows. It is not currently made automatically for any system. It is covered by its own licence; see the `README` and `NOTICE` files in that directory.

Limbo compiler

The compiler now adds a source file name (relative to the Inferno root) to each Dis file, to allow *debug*(2) and thus the debugger `wm/deb` and other commands such as *profile*(1) and *stack*(1) to find source and `.sbl` files without prompting.

The initialisation of large arrays avoids deep recursion, preventing a trap on Nt and a large stack on other platforms.

Overflow is avoided when sorting integers for case statements.

Compilers and architectures

The linker `5l` has a critical bug fix in `utils/5l/span.c` that fixes a bug in the flushing of literal pools.

`5coff` has a small change to make the output conform to actual practice not COFF documentation.

Hosted and Native Inferno

Emu has the following fixes and improvements:

Trap handling on Windows now (we hope) does all that is required to work on many versions, variants, updates and releases.

The cursor appears correctly under Windows 2000.

Windows *emu* passes page up, down scroll, pause, insert, delete and print characters through to `/dev/keyboard`.

`styx.c` prevents bad Styx messages from causing trouble.

`devenv.c` returns `file exists` if an attempt is made to create an existing name; it implements `ORCLOSE`.

`devroot.c` makes directories mode 555 not 777.

For the native kernels only:

Env(3) is now provided for native kernels. To add it to a kernel, change the kernel configuration file as follows:

1. Include the device driver `env` in the `dev` section.
2. Include the support file `env` in the `port` section.
3. Include the name `/env` in the `root` section.

To exclude it from a kernel, include the support file `noenv` in the `port` section. You should only do this if you are trying to make a small highly specialised kernel; general applications are likely to make more use of `/env` now that it is there, eventually to replace `sysenv` and to select locales.

Common floating-point emulator code has moved from platform-specific directories to `/os/port/fpi.c` and `/os/port/fpimem.c`, with corresponding changes to configuration files and `mkfiles`.

The scheduling code in `/os/port/proc.c` has changed to support wait-for-interrupt. If no process can be scheduled, the platform-specific function `void idlehands(void)` is called, with interrupts *off* (unlike the function of the same name in Plan 9). On most platforms, it currently is an empty function defined by `#define` in `fns.h`, and the scheduler effectively spins waiting for an interrupt to make a kernel process ready, but on the iPAQ and a few other platforms it uses the hardware-specific ‘‘wait for interrupt’’ function, for power saving.

A new package `ipaq` is available that populates `/os/ipaq` with the preliminary Inferno port to the Compaq iPAQ.

`/os/ip` has incorporated bug fixes and improvements from Plan 9 to keep the source code up to date:

`/net/ndb` has been added, to allow for future changes in IP configuration code

permissions are checked more carefully; `wstat` is implemented

`Conv` structures are now unlocked on `close` by `devip.c` not by each protocol's implementation; the `car` lock for connect/announce no longer exists, because the conversation itself is locked

some missing `waserror` calls have been added

ensure local port is unique across existing conversations

`tos` can be set for a converstation by a `tos` control message, and is retained during routing

`qdiscard` in `qio.c` returns the number of bytes discarded

protocol handlers `esp.c`, `gre.c`, `icmp.c`, `ip.c`, `ipifc.c`, `ipmux.c`, `rudp.c` and `tcp.c` have consequentially changed; the TCP/IP implementation most extensively; `il.c` and `udp.c` have not yet been realigned with Plan 9

medium drivers use the structure-member initialisation extension of Plan 9 C, to insulate driver source text from changes in the layout of the `Medium` structure

The SA1100 UART driver now correctly pushes input up the stack when the FIFO empties.

The remaining points are common to both hosted and native Inferno:

Cons(3) implements the file `kprint` to capture Inferno console messages; and a file `jit` that can be used to set the compile-on-the-fly option dynamically or read its current state.

There is a new kernel function:

```
char* seprint(char *buf, char *ebuf, char *fmt, ...);
```

which puts a formatted result into `buf` never writing beyond `ebuf-1` (including the trailing null byte). It returns the address of the next available byte in *buf*.

`kfs`'s `Eexist` error has become `Eexists` to remove a clash with the new `Eexist` name in `error.h`

`exportfs.c` maintains offsets in exported directories correctly.

The undocumented `devaudit.c` has been removed.

Some Limbo profiler bugs have been fixed.

A race for the use of a shared semaphore has been fixed in `devprog.c`.

`devprog.c` has a new debugging event: `load` *filename* corresponds to the execution of a Dis `load` instruction.

`devdraw.c` implements `readpixels` from a window

2. Updating the software

You should install the updates as the host operating system user who owns the Inferno files and directories on your system. You might like to take a backup copy of the existing tree, just in case. Do the following to update the installation.

1. Fetch the update archives required, namely `inferno.tgz`, `src.tgz`, `utils.tgz` and any platform-specific packages required for your installation. If you are running Windows, for instance, you will need `Nt.tgz`; if running Plan 9, you will need `Plan9.tgz`. These are gzip'd tar files containing files starting with the directory name `updates/20011003`. Unpack each in your Inferno root directory. For instance, you can unpack `inferno.tgz` using the Inferno commands:

```
cd /
gunzip <inferno.tgz | gettar
```

The following instructions assume they are visible in the Inferno hierarchy. Make sure there is sufficient space in the file system holding that hierarchy. Each archive can be removed after unpacking, and the `updates` directory can be removed after installation.

2. After unpacking the archives, (re)start the existing Inferno *emu*; it will be quicker if you use the `-c1` option to force compiled mode. It is best to use only the Inferno console; do not start the window system, since the updates will change files in the running system. On the other hand, it is a good idea to make the window in the host operating system a scrolling one, so that you can scroll back to see any errors. All following commands are run in the Inferno environment.

3. Change to the directory containing the updates:

```
cd /updates/20011003
```

4. Updated installation software was included in `inferno.tgz`. You must first unpack that installation software, as follows:

```
sh ./unpacktools
```

5. Update the installed Inferno, source and utility source directories using the script `applybase`:

```
sh ./applybase
```

That script updates the `inferno`, `src` and `utils` packages. You might see warnings if you have modified any non-configuration files from the original release.

6. Update one or more platform specific files for your platform(s). The directories are named after the platforms: `Solaris` for Solaris, `Plan9` for Plan 9, `Nt` for all Windows systems, and so on. For each *platform* that you run, do:

```
sh ./applyplat platform
```

For instance, if you use Windows, run

```
sh ./applyplat Nt
```

The iPAQ distribution is installed the same way (it is just another platform):

```
sh ./applyplat ipaq
```

which populates `/os/ipaq` in the Inferno tree.

6. Quit *emu*. The new version of *emu* will be called `emu.new` in the platform-specific directory (eg, `Solaris/sparc/bin/emu.new`). Rename the old `emu` file as `emu.old`, then rename the new `emu.new` as `emu` on Plan 9 and Unix systems, or `emu.exe` on Windows. When run, it should announce itself as ‘‘Inferno Third Edition (3 October 2001)’’.