

# Chez Scheme Version 7.1 Release Notes

## Copyright © 2006 Cadence Research Systems

### All Rights Reserved

### August 2006

## 1. Overview

This document outlines the changes made to *Chez Scheme* for Version 7.1 since Version 7.0.

Version 7.1 is available for the following platforms:

- Intel 80x86 Linux
- Intel 80x86 Linux, threaded
- Intel 80x86 Windows 95/98/ME/NT/2000/XP
- Intel 80x86 Windows 95/98/ME/NT/2000/XP, threaded
- Apple PowerPC Mac OS X 10.4
- Intel 80x86 FreeBSD
- Intel 80x86 FreeBSD, threaded
- Sun Sparc Solaris, 32-bit
- Sun Sparc Solaris, 32-bit threaded
- Sun Sparc Solaris, 64-bit
- Sun Sparc Solaris, 64-bit threaded

This document contains three sections describing significant (1) functionality changes, (2) bugs fixed, and (3) performance enhancements. A version number listed in parentheses in the header for a change indicates the first minor release or internal prerelease to support the change.

More information on *Chez Scheme* and *Petite Chez Scheme* can be found at <http://www.scheme.com>, and extensive documentation is available in *The Scheme Programming Language, 3rd edition* and the *Chez Scheme Version 7 User's Guide*.

## 2. Functionality Changes

### 2.1. FreeBSD 'x86 support (7.1)

Support for running *Chez Scheme* under FreeBSD on Intel 'x86 architectures has been added. The non-threaded version uses the machine type "i3fb," and the threaded version uses the machine type "ti3fb."

### 2.2. Exec shield support (7.1)

Support for running *Chez Scheme* under Linux kernels with the "exec shield" feature enabled has been added.

## 2.3. `quasisyntax` (7.1)

New `quasisyntax`, `unsyntax`, and `unsyntax-splicing` syntactic forms have been added. A `quasisyntax` is ilke as `syntax` form except that the portions encapsulated within an `unsyntax` or `unsyntax-splicing` form are evaluated and their values inserted into the output, as with `quasiquote`. Hash-backquote (`#'`), hash-comma (`#,` ), and hash-comma-at (`#,@`) abbreviations may be used by analogy with the similar `quasiquote` abbreviations.

## 2.4. `syntax->vector` (7.1)

The new procedure `syntax->vector` takes a syntax object representing a vector-structured form and returns a vector of syntax-objects, each representing the corresponding subform of the input form, in a manner similar to the existing `syntax->list` procedure.

## 2.5. MacOS X `dlopen` support (7.1)

The foreign interface, including `load-shared-object`, now use the `dlopen` C library function and related features recently added to MacOS X for loading foreign code and looking up foreign entry points.

## 2.6. Universal `foreign-callable` support (7.0a)

Support for `foreign-callable` in Version 7.0 and prior releases was limited to the Intel Windows and Linux environments (threaded and nonthreaded). `foreign-callable` is now supported for the threaded and nonthreaded Solaris (32- and 64 bit) and PowerPC MacOS X.

## 2.7. New `command-line` parameter (7.0a)

The existing `command-line-arguments` parameter is set to a list of the command-line arguments by the default value of the `scheme-script` parameter whenever a Scheme shell script is run. The new `command-line` is similar, but is set to include as well the name of the script as the first element. Thus, `(car (command-line))` can be used to determine the script, and `(cdr (command-line))` can be used to determine the command-line arguments.

## 2.8. `Socket` example (7.0a)

The socket example found in `examples/socket.ss` in the release directory has been made more robust. The updated code also appears in the *Chez Scheme Version 7 User's Guide*.

# 3. Bug Fixes

## 3.1. Multiple-value handling bug (7.1)

A bug in the inliner's handling of `call-with-values` that could result in an invalid memory reference at higher optimization levels has been fixed.

## 3.2. Removed `open-input-file` `exclusive` option (7.1)

The `exclusive` option has been eliminated from the file open operations, including `open-input-file`, since the underlying mechanism used on many operating systems does not support exclusive access to read-only

files. The option remains for output and input/output files.

### 3.3. Unbound meta variables (7.1)

A bug that caused variables defined with `meta define` to be unbound in environments other than the interaction environment has been fixed.

### 3.4. Unbound compiler-related variables (7.1)

The variables `make-boot-header` and `compile-script` previously evaluated to the value `#<unbound>` in *Petite Chez Scheme*. They are now bound to procedures that report that the compiler is not loaded, as with other compiler-related variables. [This bug dated back to Version 6.9c.]

### 3.5. Logical test operations (7.0a)

A bug in the optimizer's treatment of `logtest`, `logbit?`, `fxlogtest`, and `fxlogbit?`, which caused it to treat their return values as always true in test contexts, has been fixed. [This bug dated back to Version 6.9d.]

### 3.6. Format “\$” bug (7.0a)

A bug that caused `format` to reject exact real numbers has been fixed, and `format` also now produces a more appropriate error message when passed a non-real number. [This bug dated back to Version 6.9b.]

### 3.7. `thread?` for nonthreaded versions (7.0a)

The `thread?` procedure, like other threading procedures, is no longer defined in the nonthreaded versions of the system. [This bug dated back to Version 6.5.]

## 4. Performance Enhancements

### 4.1. Record accessors (7.1)

The speed of record field accessors and mutators at optimization levels 2 and below has been improved by inlining the actual memory loads and stores.

### 4.2. `foreign-callable` for nonthreaded versions (7.0a)

The speed of a call into Scheme via `foreign-callable` has been improved slightly for nonthreaded versions of the system. The difference is likely to be noticeable only for calls to Scheme procedures that execute quickly.