

# bencode.scm

## BitTorrent Bencode Decoding in Scheme

Version 0.1  
2005-04-17

Neil W. Van Dyke  
neil@neilvandyke.org

<http://www.neilvandyke.org/bencode-scm/>

Copyright © 2005 Neil W. Van Dyke. This program is Free Software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but without any warranty; without even the implied warranty of merchantability or fitness for a particular purpose. See <http://www.gnu.org/copyleft/lesser.html> for details. For other license options and consulting, contact the author.

## 1 Introduction

`bencode.scm` parses the *bencoding* format of the BitTorrent network protocol into basic Scheme data types (and currently PLT-specific byte strings). This is useful for inspecting `.torrent` files, and might later be used in the implementation of a BitTorrent client or protocol analyzer.

The format interpretation is based on the undated [BitTorrent protocol documentation Web page](#) as viewed on 2005-04-17. The mapping from those bencoding types to Scheme types is:

*String*      PLT byte string.

*Integer*     Scheme integer.

*List*        Scheme list.

*Dictionary*

Scheme list with the symbol `dictionary` as its head, and an association list as its tail.

For example, a parse of a certain real-world `.torrent` file:

```
(unbencode (open-input-file "debian.torrent"))
⇒
((dictionary
  ("announce" . #"http://cdimage.debian.org:6969/announce")
  ("comment" . #"Debian CD from cdimage.debian.org")
  ("creation date" . 1105009474))
```

```
(#"info"
 dictionary
  ("length"      . 600158208)
  ("name"       . #"debian-30r4-i386-binary-1.iso")
  ("piece length" . 524288)
  ("pieces"     . [...large byte string...]))))
```

`bencode.scm` is currently specific to PLT 299/3xx, due to the need for byte I/O and some representation for byte sequences. Otherwise, the code has been written to require only R5RS, SRFI-6, and SRFI-23.

## 2 API

`unbencode-single` *port* [Procedure]

Parses a single bencoding object (and any child objects, in the case of a list or dictionary) from input port *port* and yields the Scheme representation.

`unbencode` *port* [Procedure]

Yields a list of the Scheme representations of all bencoding objects parsed from input port *port*.

## 3 Tests

The `bencode.scm` test suite can be enabled by editing the source code file and loading [Testeez](#).

## History

Version 0.1 — 2005-04-17  
Initial release.