The Icon Program Library; Version 9.3

Ralph E. Griswold and Gregg M. Townsend

Department of Computer Science
The University of Arizona
Tucson, Arizona

IPD282
November 25, 1996
http://www.cs.arizona.edu/icon/docs/ipd282.htm

Note: This is an abbreviated description without a contents listing. The full version is given in IPD279.

1. Introduction

The Icon program library consists of Icon programs, procedures, documentation, and data. Version 9.3 of Icon is required for some parts of the library of the library [1,2].

2. Library Reorganization

With this release of the Icon program library, we are continuing the reorganization of library procedures into modules by topic. If you have been using an earlier version of the Icon program library, you may need to make some changes to link declarations in your programs. If you get error messages because of missing files, check the following modules to locate the procedures you need:

convert	type conversion and formatting procedures
datetime	date and time procedures
factors	procedures related to factoring and prime numbers
io	procedures related to input and output
lists	list manipulation procedures
math	procedures for mathematical computation
numbers	procedures for numerical computation and formatting
random	procedures related to random numbers
scan	scanning procedures
sets	set manipulation procedures
sort	sorting procedures
strings	string manipulation procedures
tables	table manipulation procedures

3. Unloading the Library

Note: The complete library, when unloaded, requires about 8.5MB of disk space. In particular, some documents in PostScript form are quite large. If your disk space is limited, take this into consideration before starting to unload.

The library is designed to be unloaded in a hierarchy that contains separate directories for different kinds of material. Material that requires graphics [2] is in separate directories whose names begin with `g`. If Icon doesn't support graphics on your platform, you can ignore these directories.

The directory structure for this version of the library is

```
        |--data                        data
        |
        |--docs                        documentation
        |
        |--incl                        include files
        |
        |--packs                       packages
        |
        |--procs                       procedures
        |
        |--progs                       programs
|--ipl--|
        |--gdata                       as above, but for graphics
        |
        |--gdocs
        |
        |--gincl
        |
        |--gpacks
        |
        |--gprocs
        |
        |--gprogs
        |
        |--cfuncs                      loadable C functions
```

The packages contain material that is too complex fit into other parts of the hierarchy or that does not conform to the library structure.

The loadable C functions are for platforms on which Icon supports the built-in function `loadfunc()`. See the `README` in that directory for more information.

The library files are packaged in different ways for different platforms. See the installation instructions for your platform.

4. Link and Include Search Paths

Many library programs link procedures. For example, `options()` is used by many programs for processing command-line options and is linked from "ucode" files obtained from translating `options.icn`.

Icon searches for ucode files first in the current directory and then in directories specified by the `IPATH` environment variable. `IPATH` consists of a sequence of blank-separated path names. The search is in the order of the names. For example, on a UNIX system running *csh*,

```
setenv IPATH "../procs /usr/icon/ilib"
```

results in a search for file names in link declarations first in the current directory, then in `../procs`, and finally in `/usr/icon/ilib`.

Files included by the preprocessor directive `$include` are searched for on `LPATH`. It has the same form as `IPATH`.

The method of setting `IPATH` and `LPATH` varies from system to system.

Since the current directory always is searched first, `IPATH` and `LPATH` need not be set if ucode and include files are placed in the same directory as the program files. See the next section.

5. Installing the Library

Installing the Icon program library consists of two steps: (1) translating the procedure files to produce ucode files and (2) translating and linking the programs.

Ucode files are produced by translating the procedure files with the `-c` option to `icont`, as in

```
icont -c options
```

which translates `options.icn`. The result is two ucode files named `options.u1` and `options.u2`. The `.u1` file contains the procedure's code and the `.u2` file contains global information about the procedure. It is these files that a link declaration such as

```
link options
```

needs.

Scripts for translating the procedure files are provided with the distribution. Once the procedure files have been translated, the ucode files can be moved to any place that is accessible from `IPATH`.

The programs are translated and linked using icont without the `-c` option, as in

```
icont deal
```

which translates and links `deal.icn`, a program that produces randomly selected bridge hands.

The result of translating and linking a program is an "icode" file. On some platforms, the name of the icode file is the same as the name of the program file with the `.icn` suffix removed (for example, `deal`). On other platforms, the icode file name has the suffix `.exe` in place of `.icn` (for example, `deal.exe`). Scripts for translating and linking the programs are provided with distributions for individual platforms. Instructions for building the programs contained in separate packages are included with those packages.

Some platforms (UNIX and MS-DOS, for example) support the direct execution of icode files. On such systems, an icode file can be run just by entering its name on the command line, as in

```
deal
```

On other systems, it is necessary to run `iconx` with the icode file as an argument, as in

```
iconx deal
```

(This also works on systems that support direct execution.) Note that the suffix (if any) need not be mentioned.

Many library programs take arguments and options from the command line. Options are identified by dashes. For example, in

```
deal -h 10
```

the `-h 10` instructs `deal` to produce 10 hands.

Icode files can be moved to any location accessible from your `PATH`. Ucode and include files are needed only during linking. They need not be accessible when icode files are run.

6. Usage Notes

It is important to read the documentation at the beginning of programs and procedures in the library. It includes information about special requirements, limitations, known bugs, and so forth.

Some of the programs in the Icon program library are quite large and may require more memory than is available on some platforms.

7. Disclaimer

The material in the Icon program library is contributed by users. It is in the public domain and can be freely copied, although author information should be left intact and any modifications should be properly attributed.
Neither the Icon Project nor the authors of material in the Icon program library assume any responsibility as to its correctness or its suitability for any purpose. The responsibility for use of the Icon program library lies entirely with the user.

8. Contents

Programs, procedures, definitions, and C functions are listed in a separate set of web pages. These pages include indices with links to detailed descriptions and source code.

The library also includes the following additional directories.

8.1 Data -- `data`

`*.csg`	`data for csg.icn`
`*.krs`	`data for kross.icn`
`*.lbl`	`data for labels.icn`
`*.rsg`	`data for rsg.icn`

`*.tok`	sample output of syntactic token counting
`*.tur`	data for turing.icn
`*.txt`	plain text
`chart.gmr`	data for ichartp.icn
`conman.sav`	data for conman.icn
`farber.sen`	"Farberisms"
`header`	skeleton header for Icon program files
`hebcalen.dat`	data read by hebcalen.dat
`hebcalen.hlp`	help file for hebcalen.dat
`hebcalpi.hlp`	data read by ProIcon version of hebcalen.dat
`icon.wrd`	English words containing the substring "icon"
`ihelp.dat`	data for ihelp.icn
`linden.dat`	input to xlinden.dat
`noci.wrd`	English words containing the substring "noci"
`palin.sen`	Palindromic sentences
`pas128.cpt`	Pascal triangle carpet to 128
`pt*.gmr`	data for pt.icn
`sample.grh`	sample data for graphpak.icn
`skeleton.icn`	skeleton used to create/update Icon programs
`termcap.dos`	termcap data for MS-DOS
`termcap2.dos`	alternative termcap data for MS-DOS
`verse.dat`	vocabulary for verse.icn

8.2 Data -- `gdata`

`*.clr`	color lists, mostly from Icon palettes as named
`*.gif`	GIF images
`*.iml`	lists of image strings
`*.ims`	image strings in Icon code format
`*.lch`	data for gpacks/tiger/tgrmap.icn
`*.pts`	data for facebend.icn
`gpxtest.gif`	GIF image from gpxtest.icn
`gxplor.dat`	test script for gxplor.icn
`linden.dat`	input to linden.icn
`uix.dat`	data for testing XIB-to-VIB conversion
`vibapp.icn`	sample VIB application
`xibapp.icn`	sample XIB application
`xnames.ed`	ed(1) script to convert 8.10 function names to 9.0

8.3 Documentation -- `docs`

`address.doc`	documentation for address procedures
`hebcalen.hlp`	documentation for hebcalen.icn
`hebcalpi.hlp`	documentation for hebcalpi.icn
`iconmake.doc`	make skeleton for Icon
`ipp.doc`	supplementary documentation for ipp.icn
`mr.man`	manual page for mr.icn

`post.1`	`manual page source for post.icn`
`polywalk.txt`	`description of polynomial programs`
`procs.pdx`	`index to procedures`
`pt.man`	`manual page for pt.icn`
`*.fdx`	`indexes to files`

8.4 Documentation -- `gdocs`

`gprocs.pdx`	`index to procedures`
`gtrace.doc`	`documentation for graphic traces`
`penelope.ps`	`PostScript documentation for penelope.icn`
`vib.ps`	`PostScript documentation for interface builder`
`vidgets.ps`	`PostScript documentation for vidgets`
`*.fdx`	`indexes to files`

8.5 Packages -- `packs`

`ftrace`	`function tracing`
`ibpag2`	`LR-based parser generator`
`idol`	`Idol; object-oriented Icon written in Icon`
`itweak`	`interactive debugger`
`loadfunc`	`C functions loaded dynamically`
`skeem`	`Scheme language, implemented in Icon`

8.6 Packages -- `gpacks`

`ged`	`window-based editor`
`tiger`	`map drawing from Census TIGER data`
`vib`	`graphics interface builder`

9. Contributions to the Icon Program Library

New material for the Icon program library always is welcome. See Reference 3 for guidelines and submission instructions.

10. Feedback

If you encounter problems with material in the Icon program library, please let us know. If you can provide corrections or improvements to library material, please send them by electronic mail or on a diskette.

We can be reached as follows:

Icon Project
Department of Computer Science
The University of Arizona
P.O. Box 210077
Tucson, AZ 85721-0077 U.S.A.

(520) 621-6613 (voice)

(520) 621-4246 (fax)

icon-project@cs.arizona.edu

Acknowledgements

Dozens of persons have contributed material to this release of the Icon program library. See the program material itself for authorship information.

References

1. R. E. Griswold, C. L. Jeffery and G. M. Townsend, *Version 9.3 of the Icon Programming Language*, The Univ. of Arizona Icon Project Document IPD278, 1995.

2. G. M. Townsend, R. E. Griswold and C. L. Jeffery, *Graphics Facilities for the Icon Programming Language; Version 9.1*, The Univ. of Arizona Icon Project Document IPD281, 1995.

3. R. E. Griswold, *Icon Program Library Submissions*, The Univ. of Arizona Icon Project Document IPD151. 1996.

Icon home page