The Standard ML Basis Manual

This book provides a description of the Standard ML (SML) Basis Library, the standard library for the SML language. For programmers using SML, it provides a complete description of the modules, types, and functions comprising the library, which is supported by all conforming implementations of the language. The book serves as a programmer's reference, providing manual pages with concise descriptions. In addition, it presents the principles and rationales used in designing the library and relates these to idioms and examples for using the library. A particular emphasis of the library is to encourage the use of SML in serious system programming. Major features of the library include I/O, a large collection of primitive types, support for internationalization, and a portable operating system interface.

This manual will be an indispensable reference for students, professional programmers, and language designers.

Emden R. Gansner is a Principal Technical Staff Member at AT&T Laboratories. Having taught at several prestigious universities, he is currently an adjunct Professor of Computer Science at Stevens Institute of Technology. He has published articles in numerous journals, such as the *Journal of Combinatorial Theory*, *Discrete Mathematics*, and *SIAM Journal of Algorithms and Discrete Methods*. He also jointly received a patent on a technique for drawing directed graphs.

John H. Reppy is an Associate Professor of Computer Science at the University of Chicago. He recently served as Associate Editor of *ACM TOPLAS* and is the author of *Concurrent Programming in ML*, also published by Cambridge University Press.

The Standard ML Basis Manual

Edited by

Emden R. Gansner
AT&T Laboratories

John H. Reppy
University of Chicago

CAMBRIDGE
UNIVERSITY PRESS

PUBLISHED BY THE PRESS SYNDICATE OF THE UNIVERSITY OF CAMBRIDGE
The Pitt Building, Trumpington Street, Cambridge, United Kingdom

CAMBRIDGE UNIVERSITY PRESS
The Edinburgh Building, Cambridge CB2 2RU, UK
40 West 20th Street, New York, NY 10011-4211, USA
477 Williamstown Road, Port Melbourne, VIC 3207, Australia
Ruiz de Alarcón 13, 28014 Madrid, Spain
Dock House, The Waterfront, Cape Town 8001, South Africa

http://www.cambridge.org

First published 2004

Printed in the United States of America

Typeface Times 10/13 pt. *System* LaTeX 2_ε [TB]

A catalog record for this book is available from the British Library.

Library of Congress Cataloging in Publication Data

ISBN 0 521 79142 1 hardback
ISBN 0 521 79478 1 paperback

Contents

Foreword

Of all modern programming languages, Standard ML has ascribed perhaps the highest priority to rigorous semantic definition. It is therefore the preferred language for many applications where rigor is important; this is notably true of tools for formal program analysis. It has also gained users who value its high degree of portability, a direct consequence of the unambiguity of its definition.

Now Emden Gansner and John Reppy have equipped SML with another essential ingredient: a library of signatures, structures, and functors which will greatly ease the programmer's task. The SML Basis Library has been long in gestation, but this has ensured that it contains the right things. Only by close cooperation with users, over a considerable period of time, can one be sure of consistency and balance in defining a library. We can therefore be confident that the Basis Library will bring SML into still wider use, and we owe warm thanks to its creators for undertaking an arduous task with skill, care, and dedication.

Robin Milner
Cambridge, July 2003

Preface

One essential for the success of a general-purpose language is an accompanying standard library that is rich enough and efficient enough to support the basic, day-to-day tasks common to all programming. Libraries provide the vocabulary with which a language can be used to say something about something. Without a broad common vocabulary, a language community cannot prosper as it might.

This document presents a standard basis library for SML. It is a basis library in the sense that it concerns itself with the fundamentals: primitive types such as integers and floating-point numbers, operations requiring runtime system or compiler support, such as I/O and arrays; and ubiquitous utility types such as booleans and lists. The SML Basis Library purposefully does not cover higher-level types, such as collection types, or application-oriented APIs, such as regular expression matching. The primary reason for limiting the scope in this way is that the design space for these interfaces is large (e.g. choosing between functors and polymorphism as a parameterization mechanism) and, unlike the case with lists and arrays, we do not have many years of common practice to guide the design. It is also the case that the SML Basis Library specification is a substantial document and expanding its scope would make it unwieldy.

The primary purpose of this book is to serve as a reference manual for the Basis Library, describing as clearly and completely as possible the types, values, and modules making up the Library. This specification is designed to serve both implementors of the SML Basis Library and users. While the specification is not formal, we have tried to make it precise and complete enough to guarantee a high degree of portability between implementations.

It is sometimes difficult to program from a reference manual; all the pieces are there but it is not clear how they fit together. For the working programmer who wants to use the Library, the book also discusses how the functions were meant to be used alone and together. Although not a tutorial, the book should assist the programmer in understanding and using the Library, clarifying when and how various structures should be used, and making the apparent arcana accessible.

There are certain roles the book does not attempt. As we've already noted, it is not a textbook, for either the Library or SML. There are already many fine books and papers teaching the joys of writing in SML, some of which address this Library as well. When dealing with the Library's interface to external software such as Unix or Windows, it assumes the reader already knows how to use them or has access to sources providing that information.

The Library is certainly not complete; there are some glaring omissions, such as a module for handling regular expressions or guidelines for internalization. It is assumed that, as needs are identified and consensus is reached on the design of a structure, new modules will be added to the Library or be standardized as a separate library. The evolution of the Library will be reflected in the online version of this document, the latest version of which can be found at

`http://standardml.org/Basis`

Overview of the book

The book is organized in three main parts: an overview of the Library, its structure and conventions; a tour of the main areas covered by the Library, providing programming tips, idioms, and examples aimed at Library users; and a set of manual pages defining the signatures and structures composing the Library.

The first three chapters form the first part. Chapter 1 presents the philosophy, principles, and rules concerning the design of the Library. It also notes the conventions used in documenting the Library. The second chapter lists all of the signatures, structures, and functions in the Library, noting their connections and whether they are optional. Chapter 3 considers those parts of the Library that are available at the top level, outside of any structure.

The following chapters describe some of the component areas, such as I/O and text handling, in more depth. These chapters discuss the common themes connecting the modules of a component, and note related assumptions and restrictions. The Library includes some elegant solutions to certain programming tasks, but these are not necessarily obvious from a bare presentation of the signatures. Thus, many of these chapters include short tutorial sections that discuss how various types and functions were intended to be used, including examples of idiomatic use.

Chapter 11 is the meat of the book, containing manual pages describing the signatures, structures, and functors specified by the Library, and their semantics. The modules are presented in alphabetical order. Generic modules, those with multiple possible implementations, are gathered under their defining signature. Thus, the `Char` structure is discussed in the `CHAR` section. Each non-generic module, those with a unique implementation, such as `Timer`, heads its own section shared with its signature. Significant substructures, for example, `Posix.IO` also rate their own sections.

During the design of the Library, the authors of the SML language have revised its definition [MTHM97], partly in response to the needs of the Library. The appendix describes some of the changes that have taken place in the SML language, especially in relation to the Library, and also notes where the Library differs from the initial basis described in the original SML definition. The back matter also provides an index of the exceptions defined in the Library.

Contributors

The main architects of the SML Basis Library are Andrew Appel, Dave Berry, Emden Gansner, John Reppy, and Peter Sestoft. In addition, the following people contributed to the design discussions and writing: Nick Barnes, Lal George, Lorenz Huelsbergen, David MacQueen, Dave Matthews, Carsten Müller, Larry Paulson, Riccardo Pucella, and Jon Thackray. This document is edited and maintained by Emden Gansner and John Reppy.

Acknowledgments

As usual in a work like this, many people have been involved in the process. We had helpful comments on the Library or this document from Peter Michael Bertelsen, Matthias Blume, Jeremy Dawson, Matthew Fluet, Elsa Gunter, Ken Larsen, Peter Lee, Neophytos Michael, Kevin Mitchell, Brian Monahan, Stefan Monnier, Chris Okasaki, Andreas Rossberg, Jon Thackray, Mads Tofte, Dan Wang, and Stephen Weeks. David Gay and Serban Jora helped with insights and pointers concerning IEEE floating-point numbers and the Windows operating system, respectively. We would especially like to thank our editor at Cambridge University Press, Lauren Cowles, whose patience has been unbounded.

This document was written using the *ML-Doc* toolkit, which is an SGML-based system for documenting SML interfaces. More information about ML-Doc can be found at

```
http://people.cs.uchicago.edu/~jhr/tools/ml-doc.html
```