输入输出

1. 简介

通过前面章节的学习，你已经可以在Scheme的交互式前端中编写并执行程序了。在本章中，我讲介绍如何输入和输出。使用这个特性，你可以从文件中读取数据或向文件中写入数据。

1. 从文件输入
   1. open-input-file，read-char和eof-object?

函数(open-input-file filename)可以用于打开一个文件。此函数返回一个用于输入的端口。函数(read-char port)用于从端口中读取一个字符。当读取到文件结尾（EOF）时，此函数返回eof-Object，你可以使用eof-object?来检查。函数(close-input-port port)用于关闭输入端口。代码片段1展示了一个函数，该函数以字符串形式返回了文件内容。

**9. Input/Output**

**1. Introduction**

You should be able to write and execute programs using interactive front end of Scheme using the knowledge provided by previous chapters.

In this chapter, I will explain about input and output. Using this feature, you can read and write data from/to files.

**2. Input from files**

**2.1.**open-input-file**,**read-char**, and**eof-object?

The function (**open-input-file** **filename**) is available to open a file. This function return a port for input. The function (**read-char** **port**) is to read a character from the **port**. As this function returns eof-object when it reaches the end of the file (EOF), you can check it by using **eof-object?**. The function (**close-input-port** **port**) is to close the input port. The [code 1] shows a function that returns file contents as string.

[code 1]

(define (read-file file-name)

(let ((p (open-input-file file-name)))

(let loop((ls1 '()) (c (read-char p)))

(if (eof-object? c)

(begin

(close-input-port p)

(list->string (reverse ls1)))

(loop (cons c ls1) (read-char p))))))