C-Scape用户界面管理系统

用 户 手 册

北京科海培训中心

C-scape 用户界面管理系统

用　户　手　册

曾 红 刚　编译

北京科海培训中心

目 录

第一章　引　言

C—scape 是用于控制 C 程序用户界面的工具，它容易学习，功能强大，灵活方便。实际上，它是一个可以生成或者改变任意文本或数据屏幕输入界面的 C 子程序库。C-scape 对初级程序员或有经验的程序员都是非常有用的。本手册说明如何使用 C-scape，并为其各部分内容提供了详尽的说明。

C-scape 使用简便。可以用类似于标准函数 printf 函数的子程序，高效快速地设计屏幕格式。C-scape 有一组专门用于生成弹出信息和菜单系统的高级函数。

C-scape 功能强大。它可以在程序中加入许多我们熟悉的特征：如窗口，图形支持、上下文相关的帮助、具有字环绕（word wrap）的文本编辑、滚动列表和数据有效性检查。

C-scape 适应性强。为了满足特殊应用需要，可以改变 C-scape 的任何部分，可以生成用户自己的字段类型、有效的子程序、文本编辑器、边框、帮助屏和菜单系统。

1.1　特点

C-scape 主要包括如下特点：

- 建立在 printf 之上的强有力的定义语言来布局屏幕格式。
- 窗口管理能自动地控制窗口的弹出及位置，管理输出到隐形窗口的信息，并能在文本和图形方式下工作。
- 定义字段和一些字段函数。通过这些函数可以确定字段内容的有效性并使之规范化，同时给出了字段的特征。
- 字段函数包括确定字段所显示数据的数据类型、数据的输入和编辑方法和数据的有效性检查。C-scape 具有许多不同的字段函数，并提供了用户建立自己的特殊字段函数的所有子程序。
- 一组用于文本编辑的子程序。这组子程序可以生成从弹出式的小编辑器（note pad）到完整的文本编辑器。
- 完善的屏幕边框。边框可以是简单的长方形，也可以是具有滚动光带和信息提示的标题边框—可对此类边框进行几乎所有的布局和操作。
- 强有力的上下文相关的帮助系统。该系统允许在任何时刻显示帮助信息。每一屏或字段都有自己独立的帮助信息，可方便地定制显示帮助信息的函数，以查看任何程序。
- 高级子程序。它由C-scape的低级子程序生成。这些子程序包括了各种菜单系统，如下拉菜单，弹出菜单，嵌套菜单，类似于提示符和信息的弹出方框子程序。

作为对软件开发的更进一步的辅助，C-scape 可以和 Look&Feel™ Screen Designer 配合使用．和 Look&Feel 配合，可以生成各种屏幕格式和交互式菜单，并可转化为兼容的 C 代码．这样做减少了用于诸如字段定义、字符位置、颜色等屏幕属性编码的输入工作，Look&Feel™ 同样可以产生、编辑映象文件，程序可以在执行时从屏幕文件中装入并执行菜单和屏幕功能，从而减少了执行文件大小，并可以在不重新编译执行文件的前提下改变程序的显示格式．

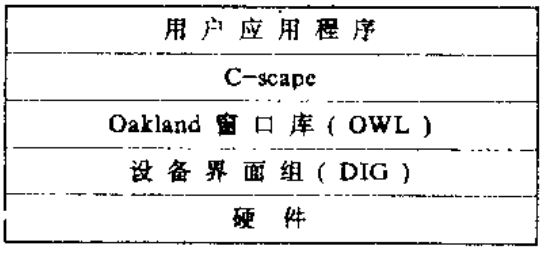
Look&Feel 可以读入 ASCII 文本文件和由 Dan Bricklins™ Demo program 所定义的幻灯片 (slide)．

关于 Look&Feel 更多的信息请参阅《Look&Feel 手册》

C-scape 可以用于许多性质完全不同的应用．如，数据库管理，通讯程序，屏幕辅助设计，spreadsheets，嵌入式处理器应用，表格布局工具，辅助安装程序，演示程序，mailing first handler，数据录入工具，源码调试及许多的其它方面的应用．

1.2 硬件需求

让我们先看一下，C-scape 在应用和物理硬件之间所处的位置．应用程序建立在 C-scape 之上，而 C-scape 是建立在 Okaland Windows Library (OWL)之上的．



OWL 处理 C-scape 与硬件系统的所有通讯工作．它也管理窗口的生成和使用工作．因为所有的 Oakland C 工具建立在 OWL 之上，所以，可以用普通的编码有效地共享显示设备．

OWL 在 DIG 中嵌入它的独立于硬件的函数，这样就保证了用 Oakland C 工具开发的程序的可移植性．DOS C-scape 建在两个标准的设备界面之上：一是直接和显示内存区通讯，二是使用了 BIOS 调用．

C-scape 的各部分是协调工作的，它面向对象，并在实现时是一致的．因此它易学，好修改，易于维护．因为字段，边框，帮助和硬件接口均使用可替代的函数指针，所以对 C-scape 的任何部分的修改是很容易的．

1.3 如何使用本手册。

C-scape 文献有两册，本手册是第一册：《C-scape 用户手册》，它给出了库的概念．

没有必要在读完本手册之后才开始工作．在读完足够的内容了解了 C-scape 的基本概念之后，就应该试着用 C-scape 编写程序．当需要仔细了解某一特征或子程序时，再参阅本手册或参考手册．

C-scape 盘中包括了一组示例程序，它们能帮助理解 C-scape 的特点．示例程序表在 DISK 1 的 Read.me 文件中．

第二章、第三章描述生成屏幕的基本问题．

如果希望更多地了解 C-scape 的内部工作，阅读第 4 章到第 8 章．

本手册的其余部分讨论 C-scape 的高级函数及其他特征．

下面是本手册内容的详细列表：

第二章：入门．

本章描述如何安装连接和编译C-scape．本章还包括了生成C-scape屏幕的实际演示．

第三章：进一步讨论．

本章更详细地讨论了生成屏幕的实际演示并引入C-scape数据对象的概念．本章还简单地讨论了高级函数的使用和屏幕定制的问题．

第四章：菜单对象．

讨论菜单对象，并说明如何生成它．在这个过程中，程序员打开和定义菜单并着重解释了 menu_printf 命令．

第五章：屏幕编辑器．(sed—Screen EDitors)

说明屏幕编辑器对象的使用及属性．本章讨论如何打开屏幕编辑器，如何定制屏幕编辑器，如何使用屏幕编辑器的一些基本特征．

第六章：字段．

本章包括有关在字段间及字段内移动的信息．也包含了如何修改和操作字段的方法．

第七章：字段函数结构．

讨论对于每一独立字段的函数的结构．本章提供一个例子，说明字段函数的特点．

第八章：标准字段函数．

讨论字段函数的各种特性，包括特殊函数，通用的数据指针（及它们的使用）和指示值．这些指示值的讨论不仅针对字段函数，在其他方面也是有用的．

第九章：高级函数．

讨论了一些建立在C-scape的子程序之上的一些高级函数．这些函数包括弹出函数，串函数，及数据和时间函数．同时还讨论了这些函数的有关使用问题．

第十章：菜单系统．

讨论C-scape的菜单系统及它们的使用．

第十一章：基本对象（bob—Basic OBject）．

本章引人了基本对象（bob）的概念．这里，我们讨论如何使用基本对象在屏幕编辑器中嵌人对象（如屏幕编辑器），说明了如何将各种对象中的一个对象嵌人屏幕编辑器的方法及嵌人特性的各种应用．

第十二章：滚动列表编辑器（sleds—Scrolling List EDitor）．

本章讨论滚动列表编辑器对象。滚动列表编辑器是一种特殊的屏幕编辑器，它可以处理变尺寸的列表。

第十三章：文本编辑。

本章描述C-scape的文本编辑能力，表述了如何生成用户自己的文本编辑子程序的方法。

第十四章：屏幕文件。

描述了C-scape如何使用由 Look&Feel Screen Designer生成的屏幕文件。

第十五章：鼠标。

本章讨论C-scape鼠标处理器，如何在应用中使用它们和有关理论问题。

第十六章：边框。

本章详细讨论了C-scape边界函数。它是第5.5节讨论的进一步深入。

第十七章：帮助系统。

本章详细叙述了C-scape的帮助系统。它表述了建立帮助信息文件和将信息与屏幕链接的方法。

第十八章：设备接口。

本章描述了C-scape提供的用于初始化显示设备，彩色，键盘和扬声器的子程序。

第十九章：更进一步的讨论。

讨论了几个问题：从C-scape 2向C-scape 3升级应考虑的因素；C-scape的限制；对外部的处理（内存分配、编码的兼容性，源码的重新编译）。

第二十章：出错处理。

描述C-scape的出错处理和出错信息的打印。

第二十一章：一般问题。

本章包含了C-scape用户经常寻问的问题。提供这些问题是希望能帮助用户尽可能快地解决应用中的问题。

第二十二章：词汇。

本章包含了C-scape术语的定义。

C-scape 第二册是《C-scape 函数参考手册》。它包括了每一 C-scape 子程序和标准字段函数以及 C-scape 错误信息表。

附录A：函数参考

本附录描述C-scape的函数及用法。

附录B：字段函数参考

提供标准字段函数的完整描述。

附录C：出错信息

出现信息表及各出错信息的描述。

关于更多的信息请参考下列 Oakland 文献：

Look&Feel 手册：本手册描述了 Look&Feel Screen Designer 的操作。

《Oakland 高级程序设计指南》：详细描述了 Oakland Window Libaray.

《C-scape Cookbook》：本书提供一组 C-scape 例子具对每一例子附注了详细的描述.

1.4 约定

C-scape 使用了如下约定:

sed:	指屏幕编辑器 (Screen EDitor).
sled:	指滚动列表编辑器 (Scrolling List EDitor).
bob:	基本对象 (Basic OBject).
ted:	指文本编辑器 (Text EDitor).
"显示器 (display)"	指实际的硬件显示设备，如终端、监视器或CRT.
"屏幕 (screen)"	指显示器上的一个字符集合，每个屏幕完成不同的功能，例如"帮助"屏、"数据输入"屏.
"菜单"	既指一个C-scape的数据对象 (菜单对象)，也指用户的选择项. 例如"123 菜单".

上述术语的具体含义可从上下文中推知.

下面是各种印刷字体的约定:

boldface	表示函数名
italics	表示变量、函数变元，及文件名.
fixed print	用于实例中的程序代码
ESC	表示键盘上的键名.

在书及 C-scape 源代码中用到下面的数据类型:

VOID *	对ANSI C编译器,用#define定义为void *，而对旧的编译器,用#define 定义为 char *. 当需要一个通用的数据指针时，用到本数据类型. C-scape 在需用到 void * 的地方均用 VOID * 代替.
SIZE_T	对ANSI C编译器,用#define定义为size_t，对旧的编译器,用#define 定义为 unsigned int.

在实例的源代码中，/* … */表示未列出的一部分代码.

第二章　启动

本章描述如何安装 C-scape 及 C-scape 盘中所包含的文件。本章还包括了实际演示，使读者熟悉 C-scape 的基本概念，从而可以设计自己的屏幕和菜单。请注意，在使用本软件之前，请阅读 1 号盘上的 Read.me 文件和 C-scape License Agreement。

2.1　安装

C-scape 软件包括很多张盘。从 1 号盘的 read.me 文件中可以得到盘内容的一览表和手册中没有给出的信息。请填写本手册前面的登记卡，并邮回。只有您邮回了登记卡，我们才给您发送源码盘。

为了将来的升级，注册登记必须合法。

在使用 C-scape 之前，必须将头文件（.h）和库文件（.LIB）从盘片上拷贝到可编译和链接的地方。这些文件及示例程序在所发行的盘上以文本方式存贮。关于释放和安装 C-scape 的有关信息请参阅 1 号盘上的 read.me 文件。

存放 C-scape 库文件和头文件最简便的地方是"cscape"子目录。

2.2　编译和链接

为了编译调用了 C-scape 函数的文件，应做如下工作：

(1) INCLUDE FILES: 应有头文件"cscape.h"

(2) NAME LENGTH: 在使用中，编译器应选择长命令选项。

(3) 其它选择：C-scape库文件是以"vanilla"标准进行编译的。使用exotic编译模式，如压缩结构，可能导致错误。

当然，对于一些专门的程序设计，可以有其它的选择及头文件。关于编译器的编译和链接命令，请参考 1 号盘的 read.me 文件。

DOS C-scape 库文件依据编译器和内存模式命名，以防止链接时偶然的不匹配性。

2.3　实际演示

本节阐述设计实用屏幕所需的内容。它将尽快地帮助读者去设计屏幕和菜单。因此，并非对每一概念都给予详述。如果出现问题，或者需了解更详细的内容，请参阅索引。下述的程序示例及其相应的注释，描述了 menu__printf 语句的语法，menu__Printf 是 C-scape 的核心。

2.3.1　menu__Printf

下面给出 C-scape 的"hello, world":

```
#include <stdio.h>
#include <cscape.h>   /* must be in all C-scape applications */

void main()
{
   menu_type menu;
   sed_type sed;
   char phone[11];

   phone[0] = '\0';

   /* Initialize the hardware interface */
   disp_Init(def_ModeText, NULL);

   menu = menu_Open();
   menu_Printf(menu, "Phone Number: %f[(###) ###-####]",
       phone, &string_func3);
   menu_Flush(menu);

   sed = sed_Open(menu);
   sed_Repaint(sed);
   sed_Go(sed);

   sed_Close(sed);

       /* shut down the hardware interface */
   disp_Close();

   printf( \nphone = )s\n", phone);
}
```

这是格式化基本的输入编辑屏所需的全部必要编码。本程序按提示进行输入，从用户处得到输入信息（让用户使用光标进行编辑），然后将结果放入变量。以上例为基础核心，可以扩充更好的字段，从而生成多种屏幕格式。

上述程序段完成一系列的操作：第一，通过调用 disp__Init 初始化 C-scape 设备接口和显示器硬件。在使用 C-scape 函数之前，必须进行初始化。第二，用菜单对象定义屏幕的布局。本菜单对象，象蓝图一样，布局了屏幕的结构，但并没有实施任何操作。用菜单生成屏幕对象和屏幕编辑器对象（菜单和屏幕编辑器的详细讨论见后），在显示器上显示并激活屏幕编辑器（sed），用户输入电话号码之后，菜单和屏幕编辑器被取消。最后，disp__Close 关闭设备接口，这是在程序结束之前必须调用函数所做的工作。

读者可以用键盘录入上述程序，也可以使用盘中提供的 sample.c 文件。同样，既可以编译和链接上述程序，也可以使用示例盘上的 example.exe（只有 DOS 版本的 C-scape 才有 example.exe 文件）。

sample.exe 可能比读者想象的要大，原因是它既包括了一些 C-scape 库函数，也包括了标准 C 的运行时间库。当加入其它的屏幕程序时，执行码的仅增加定义新屏幕格式所需要的量。

sample.exe 将下菜单显示在屏幕上：

```
phone Number: (  )
```

菜单有两个数据类型：文本缓冲区和字段阵列．文本缓冲区包含屏幕部分的正文，字段阵列包含了屏幕部分．

一个字段具有两种不同类型的位置：可写和不可写．在电话号码示例中，括号和连字符（dash）是不可写位置，空格（在源码中以#标识）是可写位置．如果在菜单内移动光标，将会发现影响打印数据的位置，但忽略了括号和连字符．

在上例中，文本缓冲区存贮了"phone Number"．根据定义，菜单的这一部分是不可写的，它仅仅包含用于提示用户进行数据输入的信息．

回顾一下我们所引入的术语：

字段：　全部的输入数据的区域

可写：　可写字符的地方．在上例中，可写处是用户可以输入电话号码数字的位置．

不可写：　字段中不可写的地方．上例中字头和电话号码之间的括号、连字符和空格是不可写的位置．

文本：　字段以外的字符．上例中字符串"phome number"是正文．

几乎所有的用于格式化屏幕的工作都由 menu__Printf 来完成：

```
menu_Printf(menu, "Phone Number: @f[(###) ###-####]", phone,
     &string_funcs);
```

menu__Printf 定义了文本并建立了字段．象"Phone Number: "通常的文本均水平输出．"@"是一个专用的格式字符，与 C 语言中的 printf 的"%"相似。这里，与 printf 中的定义数字串的"%d"一样，"@f"开始字段定义．

括号中的文本叫做"字段说明"，它定义了字段的外貌．字符"#"说明可写位置，其它字符说明不可写位置．

每一字段有两个参量：第一是存贮用户输入信息的变量；第二是函数结构的指针，它确定了字段的一般形式．

上例中，字段变量是串变量 phone，函数结构是 string__funs；说明本字段是串类型．

变量 phone 必须有足够的空间用于存放所有的字符（包括终结符"\0"）．本例中，phone 为 11 个字符长，（字符说明中每一个#对应一个，加上一个终结符"\0"）．类似于 C 语言函数，C-scape 并不检测变量的存贮空间的尺寸．如果对串变量没有提供足够的存贮空间，只要输入的信息多于存贮空间的尺寸；字符将会越界．

象 printf 一样，字段定义和通常的文本可以任意地混合．在同一行中可以有许多的字段定义．例如，下面是一个接受姓名和电话号码的 menu__Printf 示例：

```
menu_Printf(menu,
     Name: @f[##############]   Phone: @f[(###) ###-####]",
     name, &string_funcs, phone, &string_funcs);
```

注意，此时控制串之后有四个参数，每字段两个．象 printf 一样，menu__Printf 取许多参量．

也可以将 menu__Printf 分开。这里是一个共同完成上述功能的 menu__Printf 示例:

```
menu_Printf(menu, "Name: %f[##############]", name,
     &string_funcs);
menu_Printf(menu, "   Phone: %f[(###) ###-####]", phone,
     &string_funcs);
```

换行字符"\n"将输出信息设置于下一行的头上。假如想把输入信息设置于不同的行，示例如下:

```
menu_Printf(menu, "Name  :%f[##############]:\n", name,
     &string_funcs);
menu_Printf(menu, "Phone :%f[(###) ###-####]:", phone,
     &string_funcs);
```

象 printf 一样，分号可以出现在 menu__Printf 格式串的任何位置，前述的例子可以重写如下:

```
menu_Printf(menu, "Name  :%f[%s]:\n", name, &string_funcs,
     "##############");
menu_Printf(menu, "%s%f[(###) ###-####]:", "Phone :", phone,
     &string_funcs);
```

注意，给 menu__Printf 传递参量的顺序取决于专用的格式化字符的排列顺序。如遇到"%"，则从参量串中读一个参量，如果遇到"@f"，两个参量将被读出，即变量和字段函数。

如果字段是非串类型的变量，则传输不同类型的变量和函数结构。例如，输入某人的年龄，使用:

```
menu_Printf(menu, "Age:  %f[###]", &age, &int_funcs);
```

就象 C 语言的 scanf 一样，必须指针指向数据存贮区域。有一些字段函数结构，char-funs，int-funts，long-funcs，double-funcs，可定义 C 语言的数据类型：(字符 cchar)，整型 (INT)，长类型和双精度 (long，double)。(关于有效的字段函数参见 8.9 节)

2.3.2 定位

用语法"@p[row(行),col(列)]"可以直接在屏幕上定位字符。这里的 row 是行号，col 是列号，(0，0) 位置是显示屏的左上角位置。下面给于一个在字段域上面有字段名的示例:

```
menu_Printf(menu, "%p[0,0]Name %p[0,40]Phone");

menu_Printf(menu, "%p[1,0]%f[################]", name,
     &string_funcs);

menu_Printf(menu, "%p[1,40]%f[(###) ###-####]", phone,
     &string_funcs);
```

2.3.3 色彩

"@a"命令可以改变用于写菜单的色彩属性。它具有一个指明新彩色的参量。该参量是新彩色属性的值。将显示设置为反象的示例如下:

```
menu  Printf(menu, "Before: Normal @a[0×70] After: Reverse") ;
```

属性参量是一个代表属性值的串。以"0x"打头的数字表示 16 进制值，其它数字为十进制值:

```
menu__Printf(menu, "Before: Normal @a[112] After: Reverse") ;
```

也可以使用"%"表示彩色属性参量。因此可以使用#define 或逻辑属性定义彩色。这里给出使用%d 的示例:

```
menu__Printf(menu, "Before: Normal @a[%d] After: Reverse", 0×07) ;
```

P15-4

以前的"@c"命令仍可以用于设置菜单颜色，所以以前的程序仍旧有效。新的应用程序应该使用"@a"语句。

2.3.4 引用(Quoting)

怎样在显示器上显示"@"符号和包含括号的字段呢？为了在菜单中打印出 C-scape 的专用控制字符，用"@"符号引用它们。例如，两个"@"符号"@@"将会打印出一个"@"符号。所以语句:

```
menu__Printf(menu, "50 pcs @@ $ 2.00) ,
```

将产生下行输出:

```
50 pcs @@ $ 2.00
```

下述字段定义:

```
menu  Printf(menu, "Date: @[[##/##/##/@]]"date, &date__funcs) ;
```

将产生如下格式的输出:

```
Date: [  /  /  ]
```

注意，为了区别终结符"]"和字段中的"]"字符，需要将"]"字符给予引用。而"["不需要引用。至此，我们有足够多的信息来设计屏幕格式，从用户处获取输入数据。关于 menu__Printf 的更多的信息，请参阅 4.2 节。

第三章　进一步的讨论

C-scape 基于面向对象的设计。每一数据对象包含并控制其单元。对于每一个数据对象，相应地有一系列函数。通过这些函数，可以生成对象、观察其内容、修改其内容和消除对象等等。本章阐述 C-scape 的数据对象，并给出一个如何利用 C-space 设计数据输入屏的示例。后几章详细描述 C-scape 的各部分内容。

3.1　C-scape 数据对象

最重要的 C-scape 对象是菜单、字段和 scd 对象。菜单对象控制屏幕的结构和格式，它象一幅蓝图，使文本数据和称之为字段的专用数据区域显示于显示屏。

例如：

PHONE NUMBER: (####) ####-####

在此例中，PHONE NUMBER 是文本，除此之外，其余的均是字段部分。字段由两个不同类型的位置组成，可写和不可写位置。本例中，括号和连字符是不可写位置，由"#"表示的位置是可写位置，用于存贮输入数据。

"合并"是既包含可写位置，又包含不可写位置的字段全部内容的串。本例中，用户输入区域码和七个数字号码之后，merge 就是：

" (617) 491-7311"

"记录"仅指可写的位置，即，实际输入的数据，本例中记录为：

"6174917311"。

每一字段被连接两个数据对象。其一是字段中存贮输入数据的变量，其二是指向字段函数的指针。字段函数管理字段的操作。它要处理击键、编辑字段中的数据、检查数据的有效性、完成字段变量值与用于显示其内容的字符串之间的转换。文本缓冲区包含有字段内的所有菜单文本。本例中文本缓冲区包含：

"PHONE Number:

菜单对象中的数据指出了文本和字段的定位信息，记载了所定义的字段的类型，从而确定了屏幕的结构和格式。

scd 对象是具有其特征的菜单版式之一。sed 象一个框架，通过它可以观察和处理菜单对象。sed 负责在显示器上显示菜单，并从其中获取数据。任何时刻，可以在屏幕上有多个 scd，而且其尺寸和位置也是可以改变的。在屏幕 sed 的尺寸小于菜单尺寸的情况