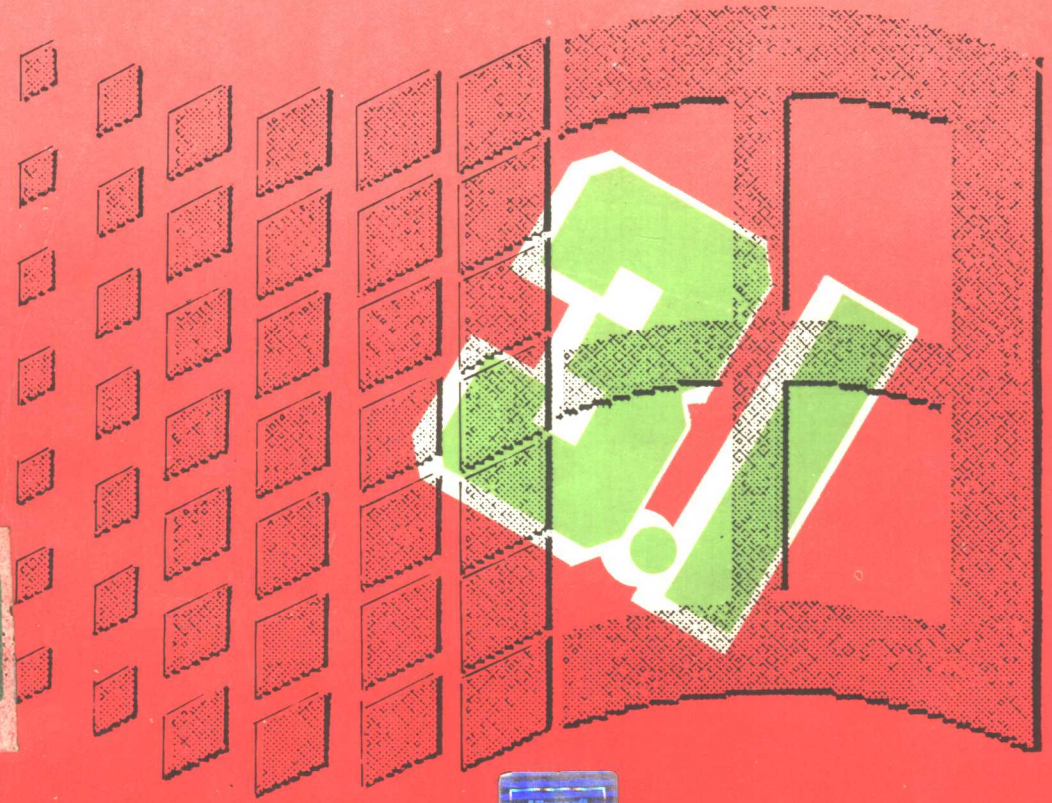
Microsoft Microsoft Microsoft

Microsoft Windows 3.1

程序员参考大全(五)

—编程工具

刘洪波 欧阳鸥 英爽 译
熊桂喜 校



清华大学出版社

Microsoft

Windows 3.1

程序员参考大全(五)
—编程工具

[美] Microsoft 公司 著

刘洪波 欧阳鸥 英 爽 译

熊桂喜 校

清华大学出版社

Microsoft Windows 3.1 Programming Tools

(京)新登字 158 号

Microsoft Windows 3.1
程序员参考大全(五)——编程工具

[美] Microsoft 公司 著
刘洪波 欧阳鸥 英 爽 译
熊桂喜 校

☆

清华大学出版社出版
北京 清华园
清华大学印刷厂印刷
新华书店总店科技发行所发行

☆

开本:787×1092 1/16 印张:11.5 字数:267 千字
1993 年 7 月第 1 版 1993 年 7 月第 1 次印刷
印数:0001—5000
ISBN 7-302-01231-8/TP・469
定价:25.00 元

引 言

Microsoft Windows 是一种用于个人计算机的单用户操作系统。它提供了大量便于开发 Windows 应用程序的工具。本书——《Microsoft Windows 3.1 程序员参考大全(五)——编程工具》将介绍如何使用这些工具。

本书的内容组织

下面简要介绍本书各章及附录的主要内容：

• 第 1 章，“创建和编辑资源”。介绍了创建和编辑 Windows 应用程序资源的三种工具。这三种工具分别是 Microsoft 图像编辑器(IMAGEDIT.EXE)、Microsoft 对话编辑器(DLGEDIT.EXE)和 Microsoft Windows 字体编辑器(FONTEDIT.EXE)。

• 第 2 章，“编译资源：资源编译器”。讲述怎样使用 Microsoft Windows 资源编译器(RC)去编译应用程序的资源并将它们加入到一个可执行的 Windows 应用程序中。

• 第 3 章，“创建 Help 文件”。讲述怎样使用 Microsoft Help 编译器、Microsoft 多分辨率位图编译器和 Microsoft Hotspot 编辑器去开发 Help 文件。

• 第 4 章，“调试：Windows 的 CodeView 调试器”。讲述怎样使用 Windows 下的 CodeView(CVW)，以便在测试 Windows 应用程序执行情况的同时查看其数据。

• 第 5 章，“高级调试器：80386 调试器”。讲述如何利用 80386 调试器(WDEB386.EXE)去调试 Windows 应用程序和动态连接库(DLLs)。

• 第 6 章，“分析系统错误：Waston 医生”。讨论如何利用 Windows 下的 Waston 医生(DRWATSON.EXE)来诊断和分析由 Windows 应用程序引起的错误。

• 第 7 章，“监视消息：Spy”。介绍了如何使用 Microsoft 的监视程序 Spy(SPY.EXE)来监视送往 Windows 应用程序中的一个或多个窗口的消息。

• 第 8 章，“监视动态数据交换活动：DDESpy”。解释如何使用 Windows 下的 DDESpy(DDESPY.EXE)去监视在 Windows 操作系统下的动态数据交换活动。

• 第 9 章，“查看堆中的内容：Heap Walker”。说明如何使用 Heap Walker (HEAPWALK.EXE)去检查在 Windows 操作系统下由 Windows 应用程序和 DLLs 占用的全局堆和局部堆。

• 第 10 章，“分析性能：Profiler”。解释如何使用 Profiler 去分析和优化在 386 增强模式的 Windows 操作系统下运行应用程序的性能。

• 第 11 章，“压缩和还原文件”。讨论如何使用文件压缩工具(COMPRESS.EXE)和文件扩展工具(EXPAND.EXE)去压缩和还原文件。

• 附录 A，“资源编译器的诊断信息”。描述了由资源编译器(RC)产生的错误信息。

• 附录 B，“Help 编译器的错误信息”。列出了 Help 编译器在建立 Help 文件时所产

生的错误信息。

·附录C，“Windows调试版本”。提供了有关Windows诊断消息的信息，来帮助用户调试Windows应用程序。

书中格式约定

下面列出了本手册所采用的用于定义语法的一些约定：

约　定	含　义
正体字	以文字方式写出的一个术语或字符的符号。例如：资源定义语句或函数名（**MENU** 或 **Create Windows**）、一条命令或命令行选项（/**nod**），用户在使用时必须照原样敲入这些术语。
斜体字	一个变量或可替换的内容。表示用户必须提供具体的值。例如：语句 **SetCursorPos**(*X*,*Y*)需要用户替换 *X* 和 *Y* 参数的值。
[ ]	方括号内表示是任选的参数。
|	表示可以选择竖杠两边列出的其中的一项。
…	表示在前面重复的术语。
BEGIN ⋮ END	代表应用程序举例中的省略部分。

另外，某些文字约定可以帮助用户理解本书的内容。

约　定	含　义
变小的大写字母	指定键名、键的次序和键的组合，例如 ALT+SPACEBAR。
标准的大写字母	指定文件名和路径、类型名和大多数的结构名（也是黑体字）以及常量名。
等宽空白	分隔例子中的代码及语句。

目 录

第 1 章

创建和编辑资源

本章介绍三个工具，可用这些工具为 Microsoft Windows 应用程序创建和编辑资源。这些工具是：Microsoft 图像编辑器、Microsoft 对话编辑器和 Microsoft Windows 字体编辑器。在 Help(帮助)中可以得到有关这些工具的完整文档。

1.1 设计图像：图像编辑器(Image Editor)

使用图像编辑器(IMAGEDIT. EXE) 可以创建图像来代表文件、窗口、光标以及 Windows 应用程序中的其它对象。图像编辑器提供了一组绘画工具，可用它们生成一些常用的图件。

图像编辑器带有上下文敏感的 Help。此 Help 中包括如何创建下列图像的帮助信息：

- 光标。它代表鼠标所指向的位置。光标也称作指示针。
- 位图。它代表静态的图像。
- 像标。它代表文件或窗口。

图 1.1 表示当用户打开一个像标文件后出现的图像编辑器窗口。

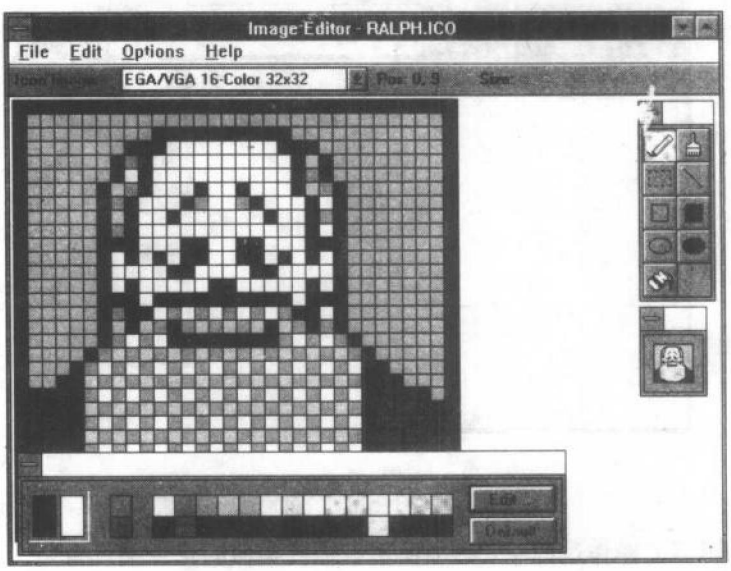


图 1.1

在使用图像编辑器时，用户必须带有一个鼠标或类似的定点类设备。

1.2 设计对话框：对话编辑器(Dialog Editor)

使用对话编辑器(DLGEDIT.EXE)，可以在屏幕上设计和测试一个对话框，以便代替在资源定义文件中所定义的 DIALOG 语句。利用对话编辑器，可以增加、修改和删除对话框里的各种控制项。对话编辑器可保存对资源定义语句所作的改变。可以把这些语句编译成二进制资源文件，以便与所编写的应用程序的可执行文件连接。

对话编辑器带有上下文敏感的 Help，此 Help 中包括下列项的信息：

- 对话编辑器中的文件操作方法。
- 显示对话编辑器窗口。
- 打开资源文件、头文件和对话框。
- 对话框的操作方法。
- 编辑各种控制。
- 对控制组的操作。
- 在资源间移动一个对话框。
- 对头文件的操作。
- 装入定制的(custom)控制。

图 1.2 表示当用户选择了 File 菜单中的 New 命令后出现的对话编辑器。

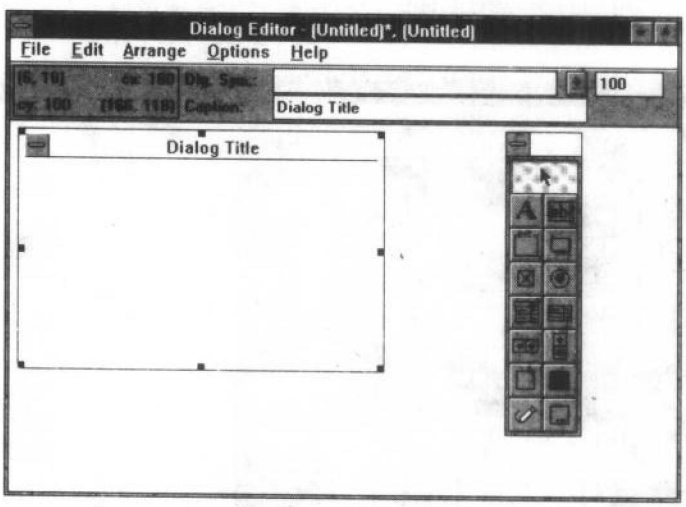


图 1.2

在对话编辑器的操作过程中，必须使用鼠标或类似的定点类设备。

1.3 设计字体：字体编辑器(Font Editor)

利用字体编辑器(FONTEDIT.EXE)，可以修改已有的字体，以便为应用程序创建新的字体。字体编辑器的 Help 描述了如何利用字体编辑器来完成下述工作：

- 编辑某个字库中的字母、数字以及其它字符。
- 修改某种字体的高度、宽度以及字符的映射方式。
- 改变字体文件头的信息。

要阅读字体编辑器的 Help 内容，可先启动 Microsoft Windows 的 Help(WINHELP.EXE)，再打开 FONTEDIT.HLP。

可以使用字体编辑器来创建或编辑光栅式字体。字体编辑器不能创建或修改向量字体或 TrueType 字体。

在用字体编辑器创建一个新的字体文件时，必须使用一个已存在的字体文件。Microsoft 在提供字体编辑器的同时提供了两个字体文件：ARTM1111.FNT 和 VGASYS.FNT。对等宽(monospace)字体，可以编辑 ARTM1111.FNT；对于变宽字体，可以编辑 VGASYS.FNT。

图 1.3 表示用户已在 Open File 对话框中打开 VGASYS.FNT 后的字体编辑器窗口。

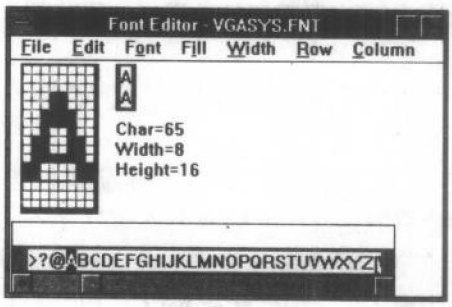


图 1.3

用字体编辑器创建一种新字体后，必须将新的字体加入到字体资源文件中。有关把一个定制字体加到字体资源文件以及如何在 Windows 应用程序中使用该资源文件的有关信息，请参考《Microsoft Windows 3.1 程序员参考大全(六)——编程指南》。

在使用字体编辑器时，必须带有一个鼠标或类似的定点类设备。

第 2 章

编译资源：资源编译器

Microsoft Windows 资源编译器(RC)是 Microsoft Windows 操作系统中的工具之一。这一章将介绍如何创建资源定义文件，如何编译应用程序的资源，以及如何把它们加进应用程序的可执行文件中。

2.1 在应用程序中包含资源

为了在 Windows 应用程序中包含资源，必须完成下列步骤的操作：

1. 为光标、像标、位图、对话框和字体创建单独的资源文件。为此，可以使用 Microsoft 的图像编辑器和对话编辑器(IMAGEDIT. EXE 和 DLGEDIT. EXE)，以及 Microsoft Windows 的字体编辑器(FONTEDIT. EXE)。

2. 创建资源定义文件。由它来描述应用程序所使用的所有资源。

3. 使用 RC 来编译资源定义文件。

4. 把编译好的资源文件加进应用程序已编译好的可执行文件中。

2.2 创建资源定义文件

在为应用程序的像标，光标、字体、位图和对话框资源创建了独立的资源文件后，必须创建一个资源定义文件。资源定义文件是一个 ASCII 文本文件，它带有扩展名. RC。

. RC 文件列出了应用程序中的每一个资源，并且最详尽地描述了某些类型的资源。对于存在于一个单独文件中的资源，如像标或光标，. RC 文件只是简单地命名此资源以及包含它的文件；而对某些资源，如菜单，则将所有的资源定义都放入. RC 文件内。

. RC 文件可以包含下列两种类型的信息：

- 语句，它命名或描述资源。
- 伪指令，它指示 RC 在资源定义文件被编译前应完成的一些动作。

下面各节描述在资源定义文件中可能会用到的语句和伪指令。有关的更详细的描述和语法，可参见《Microsoft Windows 3.1 程序员参考大全(四)——资源》。

2.2.1 单行语句

一个单行的资源定义语句可以用下列任意一个关键词开始：

关键词	说　明
BITMAP	通过命名来定义一个位图，并且指定包含它的文件的名字(为使用某个特定的位图，应用程序可通过名字进行申请)。
CURSOR	通过命名来定义一个光标，并且指定包含它的文件的名字(为使用一个特定的光标，应用程序可通过名字进行申请)。
FONT	指定包含一种字体的文件的名字。
ICON	通过命名来定义一个像标，并且指定包含它的文件的名字(为使用一个特殊的像标，应用程序可通过名字进行申请)。

2.2.2 多行语句

一个由多行组成的资源定义语句可以用下列任意一个关键词开始：

关键词	说　明
ACCELERATORS	定义菜单加速键。
DIALOG	定义一个模板，应用程序可以利用它来创建对话框。
MENU	定义一个应用程序菜单的外观和功能。
RCDATA	定义数据资源。利用数据资源，可以直接在可执行文件中包含二进制数据。
STRINGTABLE	定义串资源。串资源是以 null 结尾的 ASCII 串。可从可执行文件中装载它。

2.2.2.1 伪指令

当需要时可以在资源定义文件中使用下列伪指令，以指示 RC 完成一定的操作或给名字赋值：

关键词	说　明
#define	通过给一个名字赋值进行定义。
#elif	标明一个条件编译块的可选子句。
#else	标明一个条件编译块的最后一个可选子句。
#endif	标明一个条件编译块的结束。
#if	如果指定的表达式为真，则进行条件编译。
#ifdef	如果指定的名字已定义，则进行条件编译。
#ifndef	如果指定的名字没有定义，则进行条件编译。
#include	在 RC 处理前把一个文件的内容拷贝到资源定义文件中。
#undef	取消指定的名字的当前定义。

2.2.2.2 资源定义文件样本

下例给出了一个.RC 文件的样本，它为一个名为 shapes 的应用程序定义资源：

```
#include "SHAPES.H"

ShapesCursor CURSOR SHAPES.CUR
ShapesIcon   ICON   SHAPES.ICO

ShapesMenu  MENU
    BEGIN
        POPUP "&Shape"
            BEGIN
                MENUITEM "&Clear", ID_CLEAR
                MENUITEM "&Rectangle", ID_RECT
                MENUITEM "&Triangle", ID_TRIANGLE
                MENUITEM "&Star", ID_STAR
                MENUITEM "&Ellipse", ID_ELLIPSE
            END
    END
```

CURSOR 语句命名应用程序的光标资源为 ShapesCursor，且指定光标文件为 SHAPES.CUR，它包含了此光标的图像。

ICON 语句命名应用程序的像标资源为 ShapesIcon，且指定像标文件为 SHAPES.ICO，它包含了此像标的图像。

MENU 语句定义了一个名为 ShapesMenu 的应用程序菜单。此菜单为带有五个菜单项的弹出式菜单。

菜单定义被 BEGIN 和 END 关键词所限定。它给出了每一个菜单项和菜单标识，当用户选择这个项时其标识将被返回。例如，菜单里的第一项为 Clear，当用户选择这一项时将返回菜单标识 ID_CLEAR。菜单标识是在应用程序的头文件 SHAPES.H 中定义的。

有关资源定义文件的更详细信息、资源语句的语法，以及如何定义用户自己的资源等内容，请参见《Microsoft Windows 3.1程序员参考大全(四)——资源》。

2.3 使用资源编译器

资源编译器(RC)提供下列功能：

·把资源定义文件和资源文件(如像标和光标文件)编译成二进制资源(.RES)文件。

·把.RES 文件和由连接器产生的可执行文件(.EXE)结合在一起，结果是一个可执行的 Windows 应用程序。

·把 Windows 应用程序标定为 Windows 3.0版和 Windows 3.1 版的应用程序。

注意：每一个 Windows 应用程序和动态连接库(DLL)必须用一个 Windows 版本号来标识。为此，应对每一个所建立的 Windows 应用程序或 DLL 使用 RC，即使它不使用资源。有关 Windows 版本的更进一步信息，请参见2.3.1.1“指定选项”一节中有关/**30**和/**31**选项的描述。

2.3.1 命令行语法

为启动 RC，应使用 **rc** 命令。在命令行中，需要指定是否编译资源，是否要把编译好的资源加到一个可执行文件中，或者是否同时要完成以上两项工作。

下一行给出了 **rc** 命令行语法：

rc [*options*] *definition-file* [*executable-file*]

下面是使用 **rc** 命令时的几种可能形式：

·若只单纯编译资源，应按以下形式使用 **rc** 命令：

rc /r [*options*] *definition-file*

使用这种形式时，RC 忽略所指示的任何可执行文件。

·若要编译一个.RC 文件，且要把结果.RES 文件加入到可执行文件中，应按以下形式使用 **rc** 命令：

rc [*options*] *definition-file* [*executable-file*]

·若要编译一个没有.RES 文件的应用程序或 DLL，应按以下形式使用 rc 命令：

rc [*options*] *dll-or-executable-file*

当使用这种形式时，文件名必须带有一个明确的.EXE、.DRV 或.DLL 的扩展名。

·若只是把一个编译过的资源(.RES)文件加入到一个可执行文件中，应按以下形式使用 **rc** 命令：

rc [*options*] *res-file*.res [*execuable-file*]

2.3.1.1 指定选项

rc 命令的 *options* 参数可以包含一个或多个下列选项：

/30

标明可执行文件将在 Windows 3.0或 Windows 3.1下运行。在缺省条件下，RC 标明可执行文件只在 Windows 3.1 下运行。

/31

标明可执行文件只在 Windows 3.1 下运行。这是缺省条件。

/?

显示一个 **rc** 命令行选项的列表。

/d

为预处理定义一个符号，可以用 #ifdef 伪指令测试它。

/e

把 DLL 全局内存的缺省位置从低于扩展内存规范(EMS)存储线(bank line)改变至高于扩展内存规范存储线。本选项对 Windows 3.1 无效。

/fe *newname*

使用 *newname* 作为.EXE 文件的名字。

/fo *newname*

使用 *newname* 作为.RES 文件的名字。

/h

显示 **rc** 命令行选项的一个列表。

/i

在搜索由 INCLUDE 环境变量指定的目录前先搜索本选项指定的目录。

/k

取消 RC 的装入优化特性。如果没有指定此选项，编译器在安排可执行文件中的段和资源时，会把所有的预装入信息放在一起。

此特性使 Windows 装入应用程序时速度会更快。如果不指定/**k** 选项，所有的数据段、不可废弃代码段、以及代码段入口点都将被预装入，除非某段和它的重定位数据超过了64K。当连接应用程序时，如果在模块定义(.DEF)文件中没有给这些段指定 PRELOAD 属性，RC 将加上 PRELOAD 属性并显示一个错误。资源和段将具有同样的段边界。此边界将尽可能地小，以缩小最终的可执行文件的大小。可以通过使用 link 命令的/**a** 选项来设置边界。

/l [im32]

通知 Windows 应用程序将按照 Lotus/Intel/Microsoft 扩展内存规范(LIM EMS)版本3.2直接使用扩展内存(expanded memory)。本选项对 Windows 3.1 无效。

/m [ultinst]

当 Windows 在 EMS 4.0内存配置下运行时，给应用程序任务的每一个实例指定一个不同的 EMS 存储区(bank)(在缺省条件下，任务的所有实例共享同样的 EMS 存储区)。本选项对 Windows 3.1 无效。

/p

创建一个私有的 DLL，它仅被一个应用程序调用。这允许 Windows 更有效地使用内存，因为仅有一个应用程序(或同一应用程序的多个实例)调用这个 DLL。例如，在大框架 EMS 内存模式中，DLL 装在 EMS 存储线以上，未使用低于存储线以下的内存。本选项对 Windows 3.1无效。

/r

通过.RC 文件创建一个.RES 文件。当不想把编译过的.RES 文件加入到.EXE 文件中时，应使用这个选项。

/t

创建一个应用程序，它仅在 Windows 保护(标准或386增强)模式下运行。如果用户试图在实模式下运行此应用程序，Windows 将显示一条应用程序不能在实模式下运行的消息。本选项对 Windows 3.1 无效。

/v

显示消息报告编译器的运行进展。

/x

当搜索头文件或资源文件时，阻止 RC 检查 INCLUDE 环境变量。

/z

阻止 RC 检查 RCINCLUDE 语句。如果没有使用 RCINCLUDE 语句，使用本选项就可以极大地加快 RC 的速度。

各种选项是大小写无关的。短横线(-)可以用来代替斜线标记(/)；也可以组合单字符选项(如果它们不再需要其它参数的话)。

2.3.1.2 指定资源定义文件

rc 命令的*definition-file*参数指定资源定义文件的名字，此文件包含要加入到.EXE 文件中的资源的名字、类型、文件名及其描述。它也可以指定编译过的.RES 文件的名字，在这种情况下，RC 把编译过的资源加进可执行文件中。

2.3.1.3 指定可执行文件

rc 命令的 *executable-file* 参数指定编译过的资源将要加入到的可执行文件的名字。如果未指定可执行文件名，RC 将使用与资源定义文件具有相同名字的可执行文件(扩展名不同)。

2.3.1.4 给可执行文件换名

rc 命令的**/fe** 选项允许用户指定最终的可执行文件的名字。下例使 MYEXE.EXE 与 MYRES.RES 相结合，产生最终的可执行文件 FINAL.EXE：

```
rc /fe final.exe myres.res myexe.exe
```

2.3.2 分开编译资源文件

在缺省条件下，RC 把编译过的资源文件加进指定的可执行文件中。有时也可能想要采取单独的步骤，先编译资源，然后再把它们加入到可执行文件中。这种方式是很有用的，因为经过初步的改进后，资源文件一般改变很少。这时把应用程序的资源单独编译，然后，在每一次重新编译.EXE 文件时，再把编译过的.RES 文件加入到可执行文件中，可以节省时间。

使用/r 选项，将会单独编译资源而不把它们加入到可执行文件中。使用此选项时，RC 编译.RC 文件，并形成一个编译过的资源(.RES)文件。

例如，下面的命令读取资源定义文件 SAMPLE.RC，并创建编译好的资源文件 SAMPLE.RES：

```
rc -r sample.rc
```

在这种情况下，RC 不把 SAMPLE.RES 加进可执行文件中。

2.3.3 为预处理程序定义名字

可以在资源定义文件中指定条件分支。此分支取决于在 **rc** 命令行中是否用**/d** 选项定义了一个术语。

例如，假设应用程序有一个弹出式菜单(Debug 菜单)，它应该仅仅在调试期间才显示。当为正常的使用而编译应用程序时，就不应包括 Debug 菜单。下面的例子给出了可以加进资源定义文件中定义 Debug 菜单的语句：

```
MainMenu MENU
BEGIN
```