Free Pascal :
Manual del usuario

Manual del usuario para Free Pascal, versión 1.0.2
1.8
Diciembre 2000

Michaël Van Canneyt
Florian Klämpfl
Traducido por Luis R. Hilario

Algunas palabras antes de empezar

El objetivo de la presentación de este material es para evaluación ya que está en desarrollo.

Esta es la cuarta presentación de los avances que se están realizando en la traducción de los manuales de Free Pascal.

Me complace decirles que gracias a este esfuerzo los manuales originales han aumentado la calidad, debido a correcciones sugeridas por este proyecto, además ya es posible instalar y configurar a FPC en español.

En esta cuarta entrega se inició el capítulo 6 que trata sobre *El IDE*.

La última versión de los manuales las puedes encontrar aquí .

Para saber más sobre FPC haga clic aquí .

Algunas palabras acerca de las traducciones

A los pequeños comentarios que se encuentran en medio de "(− −)" se les debe prestar atención ya que son: traducciones alternativas que pueden ser usadas, cosas inconclusas, recomendaciones, errores de la documentación en inglés (la original), etc.

Si usted piensa que *x* traducción alternativa es mejor que la que se está usando, que existen errores en las traducciones (claro que sí), entonces envíeme un correo a la siguiente dirección: fpc@luis−digital.8m.com

Si deseas unirte es fácil, sólo tienes que escribir al correo anterior y te asignaré la cantidad de páginas que desees (de 1 hasta 3 por asignación), así de fácil. Su nombre o nickname aparecerá en la lista de contribuidores.

Su ayuda es muy apreciada, gracias.

Luis R. Hilario

Presentación #4 (06/08/2001)

Contenidos

Contenidos

1 Introducción

1.1 Acerca del documento

Este es el manual del usuario para Free Pascal. Describe la instalación y uso del compilador de Free Pascal sobre las diferentes plataformas soportadas. Este no atenta dar una lista exhaustiva de todos los comandos soportados, ni una definición del lenguaje Pascal. Mire la guía de Referencia para estas cosas. Para una descripción de las posibilidades y los funcionamientos internos del compilador, vea la guía del Programador. En los apéndices de este documento usted encontrará las listas de las palabras reservadas y los mensajes de error del compilador (con las descripciones).

Este documento describe el compilador tal como es/funciona al momento de escribir ésto. Puesto que el compilador está bajo desarrollo continuo, algunas de estas cosas descritas aquí pueden ser anticuadas. En caso de duda, consulte los archivos README, distribuidos con el compilador. Los archivos README son, en caso de conflicto con este manual, autoritarios.

1.2 Acerca del compilador

Free Pascal es un compilador de 32 bits para los procesadores [1] i386 y m68k. Actualmente, soporta 7 sistemas operativos:

- DOS
- LINUX
- ATARI (sólo la versión 0.99.5)
- AMIGA (sólo la versión 0.99.5)
- WINDOWS 32−BIT
- OS/2 (usando el paquete EMX, también trabaja sobre DOS/Windows)
- FreeBSD (usable, pero aún bajo desarrollo).

Free Pascal está diseñado para ser, tanto como sea posible, compatible con los fuentes de Turbo Pascal 7.0 y Delphi 5 (aunque esta meta todavía no se ha alcanzado), pero también mejora estos lenguajes con elementos como *function overloading (−sobre carga de funciones−*). Y, a diferencia de estos antecesores, soporta múltiples plataformas.

[1] Se comienza a trabajar para portarlo a la arquitectura ALPHA

También se diferencia de ellos en el sentido que usted no puede usar unidades compiladas desde un sistema para otro.También, al momento de escribir ésto, hay sólo una versión Beta temprana de un Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés) disponible para Free Pascal.Free Pascal consiste de tres partes:

1. El programa compilador en sí mismo.
2. La Librería de Tiempo de Ejecución (RTL, por sus siglas en inglés).
3. Programas utilitarios y unidades.

De éstos usted sólo necesita los dos primeros, para poder utilizar el compilador. En este documento, describimos el uso del compilador. La RTL se describe en la guía de Referencia.

1.3 Consiguiendo más información.

Si la documentación no da una respuesta a sus preguntas, usted puede obtener más información en Internet, en las siguientes direcciones:

- http://www.freepascal.org/ es el sitio principal. Contiene también útiles direcciones de correos y enlaces a otros lugares. También contiene las instrucciones para inscribirse a la lista de correo.
- http://www.brain.uni−freiburg.de/~klaus/fpc/fpc.html es una réplica del sitio principal de información Free Pascal.

Ambos lugares pueden ser usados para descargar la distribución de Free Pascal, aunque usted puede también encontrarlo probablemente en otros lugares.

Finalmente, si usted piensa que algo debería ser agregado a este manual (totalmente posible), por favor no vacile y contácteme en michael@freepascal.org .

Déjeme saber sobre algo útil.

2 Instalando el compilador

2.1 Antes de la instalación : Los requerimientos

2.1.1 Requerimientos del sistema

El compilador necesita al menos el hardware siguiente:

1. Un procesador I386 o superior. No se requiere de un coprocesador, aunque esto relentizará el rendimiento de sus programas si usted hace cálculos de punto flotante.

2. 4 MB de memoria libre. Bajo DOS, si usted usa el administrador de memoria DPMI, tal como bajo Windows, usted necesitará al menos 16 MB.

3. Al menos 500 Kb. de espacio libre en el disco.

2.1.2 Requerimientos del software

Bajo DOS

La distribución del DOS contiene todos los archivos que usted necesita para ejecutar el compilador y compilar programas de Pascal.

Bajo Linux

Bajo Linux usted necesita tener los programas siguientes instalados:

1. GNU as, el ensamblador de GNU.

2. GNU ld, el enlazador de GNU.

3. Opcionalmente (pero altamente recomendado): GNU make . Para fácil recompilación del compilador y la Librería de Tiempo de Ejecución, este es necesitado.

Con excepción de eso, Free Pascal debe ejecutarse en casi cualquier sistema Linux I386.

Bajo Windows

La distribución de Windows contiene todos los archivos que usted necesita para ejecutar el compilador y compilar programas de Pascal. Sin embargo, puede ser una buena idea instalar las herramientas mingw32 o las herramientas de desarrollo cygwin . Enlaces hacia ambas herramientas se pueden encontrar en http://www.freepascal.org

Bajo OS/2

A pesar de que la distribución de Free Pascal viene con todas las herramientas necesarias, es una buena idea instalar la extensión EMX para compilar y ejecutar programas con el compilador de Free Pascal. Las extensiones EMX pueden ser encontradas en: http://www.leo.org/pub/comp/os/os2/leo/gnu/emx+gcc/index.html

2.2 Instalando el compilador.

La instalación de Free Pascal es fácil, pero es dependiente de la plataforma. Discutimos el proceso para cada plataforma separadamente.

2.2.1 Instalando bajo DOS o Windows

Pasos obligatorios para la instalación.

Primero, debe conseguir los archivos de la última distribución de Free Pascal. Vienen como archivos *zip (−comprimidos .zip−)* , el cual usted debe primero *unzip (−descomprimir−)*, o puede descargar el compilador como una serie de archivos separados. Esto es especialmente útil si usted tiene una conexión lenta, pero es también adecuado si quiere instalar solamente algunas partes de la distribución del compilador. El archivo zip de la distribución contiene el programa INSTALL.EXE de instalación. Debe ejecutar este programa para instalar el compilador.

La pantalla del programa de instalación luce igual que la figura 2.1 .

El programa le permite seleccionar:

- Qué componentes quiere instalar. Por ejemplo: si quiere los fuentes o no, los documentos o no. Las partes que usted no descargo cuando hizo la descarga como archivos separados, no estarán disponibles, es decir no puede seleccionarlas.

- Dónde quiere instalarlo (la localización por defecto es C:\PP).

En orden de ejecutar a Free Pascal desde cualquier directorio sobre su sistema, usted debe extender su variable de ruta para que contenga el directorio C:\PP\BIN. Usualmente esto se hace en el archivo AUTOEXEC.BAT . Debe de lucir parecido a esto:

```
SET PATH=%PATH%;C:\PP\BIN
```

(Una vez más, si se asume que usted lo instaló en la localización por defecto).

Si quiere usar los manejadores gráficos usted debe modificar la variable de entorno GO32. Las instrucciones para hacerlo pueden ser encontradas en la documentación de la unidad Graph , en el procedimiento InitGraph .

Instalación opcional: La emulación del coprocesador

Para las personas que tienen un tipo de CPU viejo, sin coprocesador matemático (i387) es necesario instalar la emulación del coprocesador, puesto que Free Pascal usa el coprocesador para hacer todas las operaciones de punto flotante.

La instalación de la emulación del coprocesador es manejada por el programa de la instalación (INSTALL.EXE) bajo DOS y Windows 32−BIT.

Figura 2.1: La pantalla del programa DOS de instalación

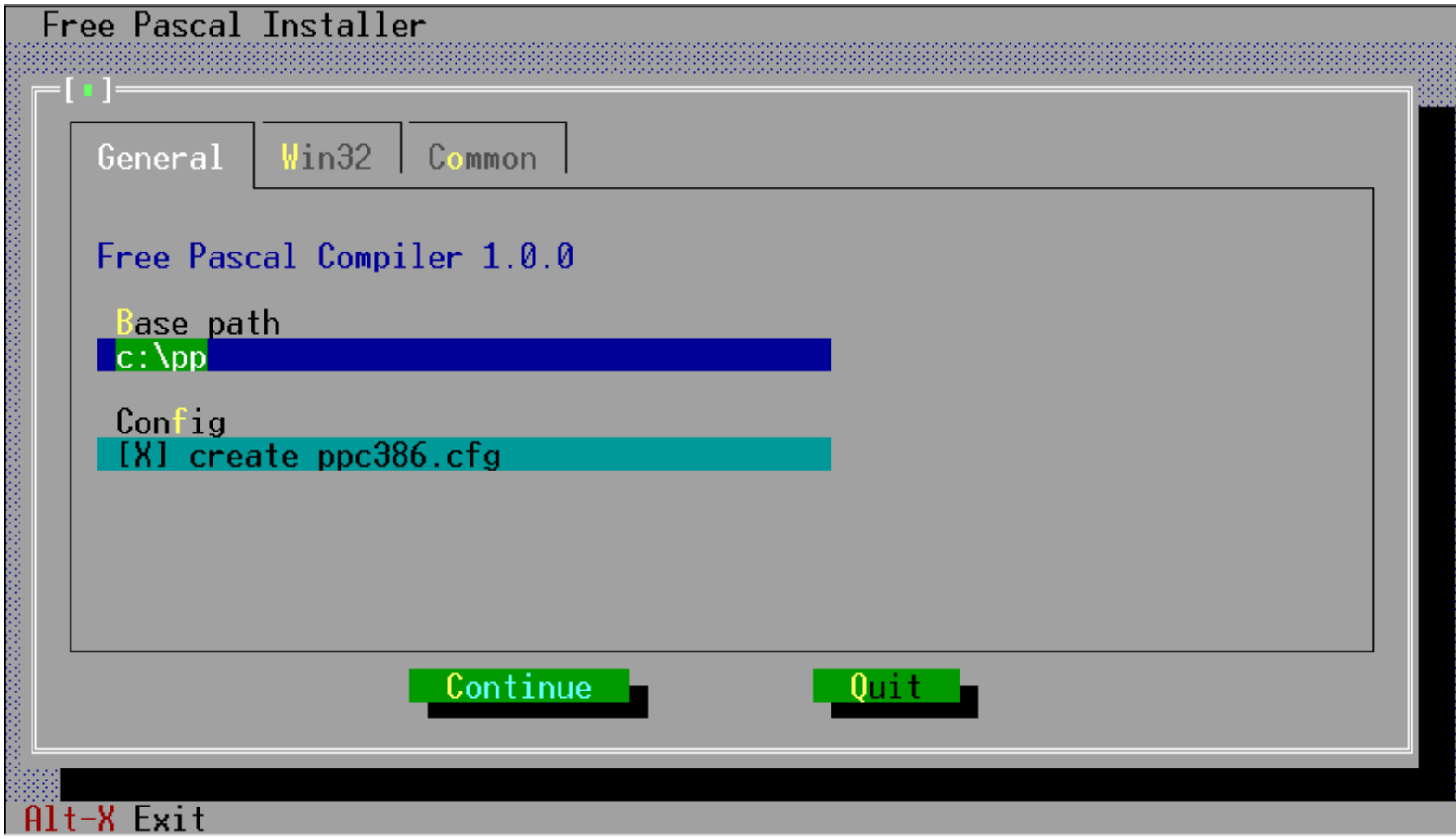
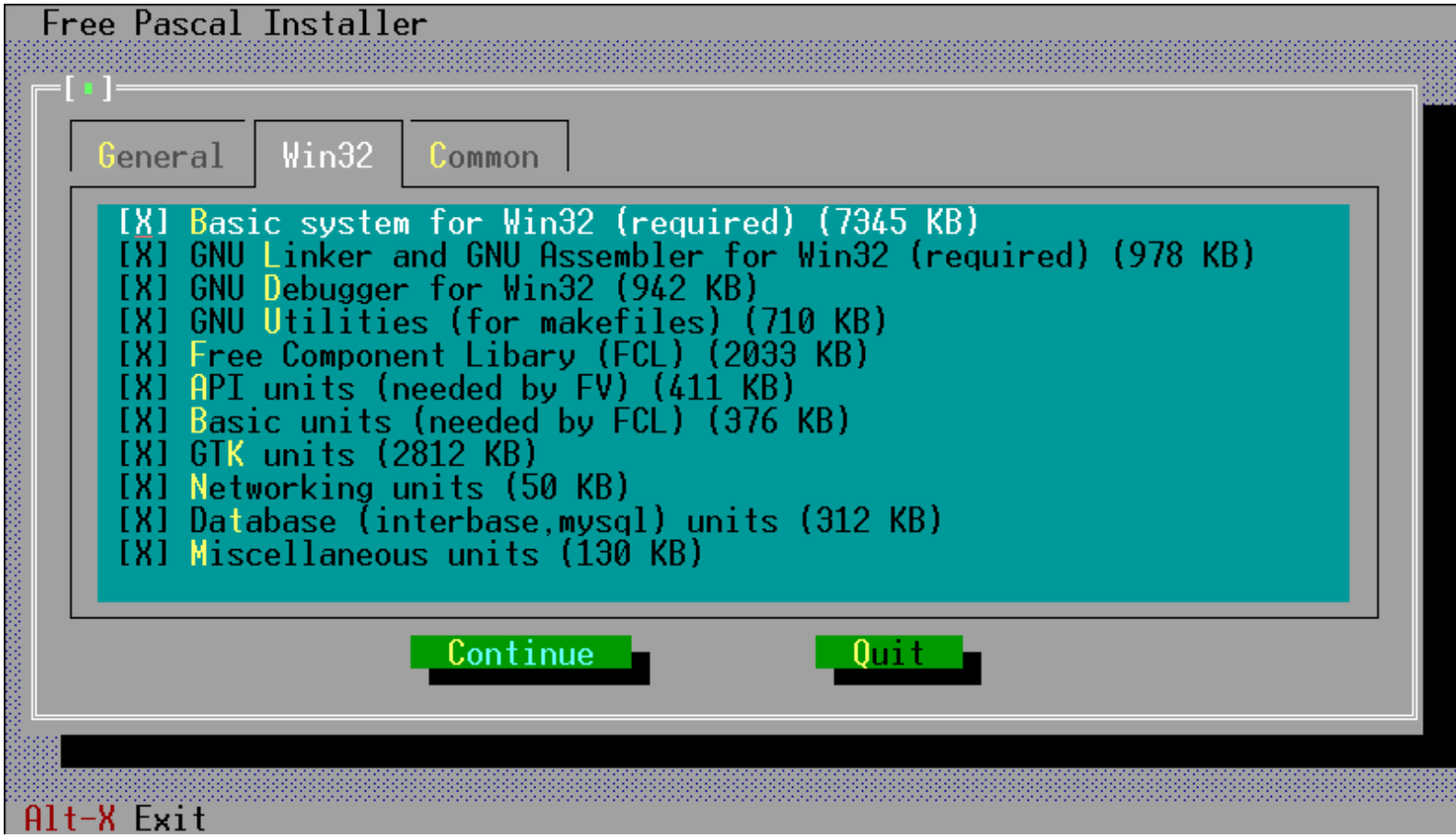


Figura 2.2: La pantalla del programa DOS de instalación



2.2.2 Instalando bajo Linux

Pasos obligatorios para la instalación.

La distribución Linux de Free Pascal viene en tres formatos:

- Una versión tar.gz, disponible también como archivos separados.
- Una versión .rpm (Red Hat Package Manager), y
- Una versión .deb (Debian). (− Error en la documentación: dice "debian" no "Debian" −)

Todos estos paquetes contienen una versión ELF de los binarios del compilador y las unidades. Los binarios más viejos aout no son más distribuidos, aunque usted todavía puede usar el compilador sobre un sistema aout si usted lo recompila.

Si usa el formato .rpm, la instalación se limita a:

```
rpm −i fpc−pascal−XXX.rpm
```

(XXX es el número de versión del archivo .rpm)
Si usa Debian, la instalación se limita a:

```
dpkg −i fpc−XXX.deb
```

Aquí de nuevo, XXX es el número de versión del archivo .deb.

Usted necesita acceso de root para instalar estos paquetes. El archivo .tar le permite hacer una instalación si usted no tiene permisos de root.

Cuando se descarga el archivo .tar, o los archivos separados, la instalación es más interactiva.

En caso de que descargara el archivo *.tar* , debe primero *untar (−descomprimir el archivo .tar−)* el archivo, en algún directorio donde usted tenga permiso de escritura, usando el comando siguiente:

```
tar −xvf fpc.tar
```

Aquí suponemos que usted descargó el archivo fpc.tar de algún lugar de la Internet. (El nombre del archivo real tendría algun número de versión en él, el cual omitimos por claridad).

Cuando el archivo sea *untarred (−descomprimido−)* , tendrá más archivos, y un programa de instalación: un archivo de comandos de instalación.

Si descargó los archivos como archivos separados, debe descargar por lo menos el archivo de comandos install.sh , y las librerías (libs.tar.gz).

Para instalar Free Pascal, todo lo que tiene que hacer ahora es dar los comandos siguientes:

```
./install.sh
```

Y entonces debe responder algunas preguntas. Son muy simples, son principalmente concernientes a dos cosas:

1. Los lugares donde usted puede instalar las diversas cosas.
2. Decidiendo si quiere instalar ciertos componentes (tales como los fuentes y los programas demos).

El archivo de comandos automáticamente detectará cuales componentes están presentes y pueden ser instalados. Ofrecerá sólo instalar los que se han encontrados. Debido a esta característica, debe dejar los nombres originales al descargarlo, puesto que el archivo de comandos cuenta con eso.

Si ejecuta el archivo de comandos de instalación como usuario root, puede sólo aceptar todos los valores por defecto de la instalación. Si no está ejecutando como root, debe de tener cuidado al proveer al programa de instalación con los nombres de directorios donde usted tiene permiso de escritura, pues procurará crear los directorios que usted especificó. En principio, usted lo puede instalar dondequiera que desee, a pesar de.

Al final de la instalación, el programa de instalación generará un archivo de configuración para el compilador Free Pascal el cual refleja las configuraciones que usted eligió. Este archivo será instalado en el directorio /etc , (si usted no está instalandolo como root , éste fallará), y en el directorio donde instaló las librerías.

Si quiere que el compilador Free Pascal use este archivo de configuración, debe de estar presente en /etc, o puede ajustar la variable de entorno PPC_CONFIG_PATH. Bajo csh, usted puede hacerlo agregando una

setenv PPC_CONFIG_PATH /usr/lib/ppc/0.99.14

línea a su archivo .login en su directorio home. (Véase también la sección siguiente).

2.3 Pasos opcionales de la configuración

Sobre cualquier plataforma, después de instalar el compilador puede desear ajustar algunas variables de entorno. El compilador Free Pascal reconoce las variables siguientes:

- PPC_EXEC_PATH contiene el directorio donde están "as" y "ld" (por defecto /usr/bin).
- PPC_GCCLIB_PATH contiene el directorio donde está libgcc.a (sin valor por defecto). Esto es sólo para Linux.
- PPC_CONFIG_PATH especifica una ruta alterna para encontrar a ppc386.cfg (por defecto bajo Linux es /etc).
- PPC_ERROR_FILE especifica la ruta y el nombre del archivo de definición de errores (por defecto /usr/lib/fpc/errorE.msg).

Estas localizaciones están, sin embargo, ajustadas en el archivo de la configuración de ejemplo el cual es creado a el final del proceso de la instalación, a excepción de la variable de PPC_CONFIG_PATH , la cual debe ajustar si usted no instaló las cosas en los lugares por defectos.

Finalmente:*(−en la documentación en inglés sólo dice " **finally** " sin mayúscula ni dos puntos−)*

También distribuido con Free Pascal un archivo README . Contiene las últimas instrucciones para instalar Free Pascal, y siempre se debe de leer primero.

2.4 Probando el compilador

Después de que la instalación este completa y las variables de entorno están ajustadas como se describió arriba, su primer programa puede ser compilar.

Incluidos en la distribución de Free Pascal están algunos programas de demostración, mostrando lo que el compilador puede hacer. Puede probar si el compilador funciona correctamente intentando compilar estos programas.

El compilador se llama:

- ppc386 bajo Linux
- PPC386.EXE bajo otros sistemas de destino.

Para compilar un programa (por ejemplo demo\hello.pp) escriba simplemente:

ppc386 hello

en el indicador de sistema. Si usted no tiene un archivo de configuración, entonces puede necesitar desirle al compilador donde puede encontrar las unidades, por ejemplo como sigue:

```
ppc386 −Fuc:\pp\units\go32v2\rtl hello
```

bajo DOS, y bajo Linux podría escribir

```
ppc386 −Fu/usr/lib/fpc/1.00/units/linux/rtl hello
```

Esto es, por supuesto, asumiendo que usted lo instaló bajo C:\PP o /usr/lib/fpc/1.00, respectivamente.

Si no consiguió un mensaje de error, el compilador ha generado un ejecutable llamado hello (sin extensión) bajo Linux, y un archivo hello.exe bajo DOS.

Para ejecutar el programa, escriba simplemente:

```
hello
```

Si todo sale bien, debe de ver el saludo amigable siguiente:

```
Hello world
```

3 Uso del compilador

Aquí describimos lo esencial para compilar un programa y una unidad. También describimos cómo hacer un ejecutable independiente del programa compilado bajo DOS. Para usos más avanzados del compilador, vea la sección sobre configurando el compilador, y el Manual del Programador.

Los ejemplos en esta sección suponen que usted tiene un ppc386.cfg el cual está configurado correctamente, y que contiene al menos la ruta ajustada para las unidades RTL. En principio este archivo es generado por el programa de instalación. Usted puede tener que chequear que está en el lugar correcto (véase la sección 5.2 para más información sobre esto).

3.1 Buscando un archivo

Antes de empezar a compilar un programa o una serie de unidades, es importante conocer dónde el compilador busca sus archivos fuentes y otros archivos. En esta sección discutimos esto, e indicamos cómo esto influye.

Comentario: El uso de barras (/) y de barras invertidas (\) como separadores de directorios son irrelevantes, el compilador lo convertirá a cualquier carácter que sea usado en el sistema operativo actual. Los ejemplos se darán usando barras, ya que esto evita problemas bajo Linux.

3.1.1 Archivos de la línea de comandos

El archivo que usted especifique en la línea de comando, tal como en

```
ppc386 foo.pp
```

será buscado SOLAMENTE en el directorio actual. Si especifica un directorio en el nombre del archivo, entonces el compilador buscará en ese directorio:

```
ppc386 subdir/foo.pp
```

buscará a foo.pp en el directorio subdir del directorio actual.

Bajo Linux, el nombre de este archivo es sensible a las mayúsculas y minúsculas, bajo otros sistemas operativos (DOS, WINDOWS NT, OS/2) este caso no se da.

3.1.2 Archivos de unidades

Cuando usted compila una unidad o un programa que necesita de otras unidades, el compilador buscará por versiones compiladas de estas unidades de la forma siguiente:

1. Buscará en el directorio actual.

2. Buscará en el directorio donde está el binario del compilador (no bajo Linux).

3. Buscará en todos los directorios especificados en la ruta de búsqueda de la unidad.

Puede añadir un directorio a la ruta de búsqueda de la unidad con la opción −Fu (véase 5.1). Cada ocurrencia de una de estas opciones insertará un directorio a la ruta de búsqueda de la unidad esto es que la última ruta en la línea de comando será buscada primero.

El compilador añade varias rutas a la ruta de búsqueda de la unidad:

1. El contenido de la variable de entorno XXUNITS, donde XX debe de ser remplazado con uno de los destinos soportados: GO32V2, LINUX,WIN32, OS2.
2. El directorio estándar de la unidad. Este directorio se determina desde la variable de entorno FPCDIR . Si esta variable no está ajustada, entonces es omitida a lo siguiente:

   - Sobre Linux:

     /usr/local/lib/fpc/VERSION o

     /usr/lib/fpc/VERSION cualquiera que se encuentre primero.
   - Sobre otros SOs: el directorio del binario del compilador, con "../ " añadido a él, si existe.

   Después de que este directorio es determinado, las rutas siguientes son añadidas a la ruta de búsqueda:

   (a) FPCDIR/units/TARGET
   (b) FPCDIR/units/TARGET/rtl

   Aquí el destino *(−TARGET−)* debe de ser remplazado por el nombre del destino para el que está compilando.

Puede ver qué rutas buscará el compilador dando a el compilador la opción −vu.

Sobre Linux, el compilador primero convertirá el nombre del archivo de una unidad a todo−minúscula. Esto es necesario, puesto que Pascal es caso−independiente, y las declaraciones Uses Unit1; o uses unit1; debe de tener el mismo efecto. También, los nombres de las unidades que son más largos que 8 caracteres primero serán buscados con su longitud completa. Si la unidad no se encuentra con ese nombre, el nombre será truncado a 8 caracteres, y el compilador buscará otra vez en los mismos directorios, pero con el nombre truncado.

Por ejemplo, suponga que el archivo foo.pp necesita la unidad bar. Entonces el comando

ppc386 −Fu.. −Fuunits foo.pp

le dirá a el compilador buscar por la unidad bar en los lugares siguientes:

1. En el directorio actual.
2. En el directorio donde el binario del compilador está (bajo Linux no).
3. En el directorio padre del directorio actual.
4. En el subdirectorio units del directorio actual.
5. En el directorio estándar de la unidad.

Si el compilador encuentra la unidad que necesita, buscará por el archivo fuente de esta unidad en el mismo directorio donde encontró la unidad. Si encuentra el fuente de la unidad, entonces comparará las fechas de los archivos (−del archivo???−). Si el archivo fuente fue más reciente modificado que el archivo de la unidad, el compilador intentará recompilar la unidad con este archivo fuente.

Si el compilador no encuentra una versión compilada de la unidad, o cuando se especifica la opción −B , entonces el compilador buscará de la misma manera por el archivo fuente de la unidad, e intenta recompilarlo.

Se recomienda ajustar la ruta de búsqueda de la unidad en el archivo de la configuración ppc386.cfg . Si hace esto, usted no necesita especificar la ruta de búsqueda en la línea de comando cada vez que desea compilar algo.

3.1.3 Archivos de inclusión

Si incluye archivos en su fuente con la directiva {$I filename} , el compilador lo buscará en los lugares siguientes:

1. Buscará en la ruta especificada en el nombre del archivo incluido (−incude???−).
2. Buscará en el directorio donde está el actual archivo fuente.
3. Buscará en todos los directorios especificados en la ruta de búsqueda del archivo incluido.

Puede añadir archivos a la ruta de búsqueda del archivo incluido con la opción −I (véase 5.1).
Como un ejemplo, considere la declaración de inclusión siguiente en un archivo units/foo.pp:

```
{$i ../bar.inc}
```

Entonces el comando siguiente:

```
ppc386 −Iincfiles units/foo.pp
```

causará que el compilador busque por bar.inc en los directorios siguientes:

1. el directorio padre de el directorio actual
2. el subdirectorio units de el directorio actual
3. el directorio incfiles de el directorio actual.

Archivos objetoCuando usted enlaza a los archivos objeto (usando la directiva {$L file.o})(−falta ")"−), el compilador buscará este archivo de la misma forma como buscó por los archivos de inclusión:

1. Buscará en la ruta especificada en el nombre del archivo objeto.
2. Buscará en el directorio donde está el actual archivo fuente.
3. Buscará en todos los directorios especificados en la ruta de búsqueda del archivo objeto.

Puede añadir archivos a la ruta de búsqueda del archivo objeto con la opción −Fo (véase 5.1).

3.1.4 El archivo de configuración

Al menos que usted especifique la opción −n (véase 5.1), el compilador buscará por un archivo de configuración ppc386.cfg en los lugares siguientes:

- Bajo Linux
    1. El directorio actual.
    2. En su directorio `HOME` , busca por .ppc386.cfg.
    3. El directorio especificado en la variable de entorno PPC_CONFIG_PATH, y bajo /etc si no está ajustada.
- Bajo todos los otros OSs:
    1. El directorio actual.
    2. Si se ajusta, el directorio especificado en la variable de entorno. PPC_CONFIG_PATH.
    3. El directorio donde está el compilador.

3.1.5 Acerca de los nombres de archivos largos

Free Pascal puede manejar nombres de archivos largos bajo Windows 32−BIT; usará el soporte para nombres de archivos largos si está disponible.

Si el soporte para nombres de archivos largos no está presente, truncará los nombres de la unidad a 8 caracteres.

No se recomienda poner unidades en los directorios que contienen espacios en sus nombre, puesto que el enlazador no entiende tales nombres de archivos.

3.2 Compilando un programa

La compilación de un programa es muy simple. Asumiendo que usted tiene un programa fuente en el archivo prog.pp, usted puede compilar este con el comando siguiente:

```
ppc386 [options] prog.pp
```

Los corchetes [ ] indican que lo que está entre ellos es opcional.

Si su archivo del programa tiene la extensión .pp o .pas, puede omitir esta en la línea de comando, por ejemplo en el ejemplo anterior habría podido escribir:

```
ppc386 [options] prog
```

Si todo sale bien, el compilador producirá un ejecutable, o, para la versión 1 del extensor del DOS, un archivo que puede ser convertido a un ejecutable.

A menos que esté usando DOS y la versión 1 del extensor del DOS, el archivo que usted obtuvo es el ejecutable. Puede ejecutarlo inmediatamente, no necesita hacer nada más. Bajo la versión 1 del extensor del DOS, se requiere un proceso adicional. En este caso véase la sección 3.5 sobre cómo crear un ejecutable.

Notará que hay también otro archivo en su directorio, con las extensiones .o. Este contiene el archivo objeto para su programa. Si compiló un programa, puede borrar el archivo objeto (.o), pero no si compiló una unidad. Entonces el archivo objeto contiene el código de la unidad, y será enlazado en cualquier programa que use la unidad que usted compiló, así que no debe removerlo.

3.3 Compilando una unidad

La compilación de una unidad no es esencialmente diferente de compilar un programa. La diferencia es principalmente que el enlazador no es llamado en este caso.

Para compilar una unidad en el archivo foo.pp , sólo escriba:

```
ppc386 foo
```

Recuerde el comentario acerca de las extensiones de archivo en la sección anterior. Cuando todo salga bien, le dejarán con 2 (dos) archivos de unidades:

1. foo.ppu Este es el archivo que describe la unidad que usted acaba de compilar.
2. foo.o Este archivo contiene el código actual de la unidad. Este archivo terminará eventualmente en los ejecutables.

Ambos archivos son necesitados si planea usar la unidad para algunos programas. No los borre. Si desea distribuir la unidad, debe proveer ambos el archivo .ppu y el .o. Uno es inútil sin el otro.

Comentario: Bajo Linux, un archivo fuente de una unidad debe tener un nombre de archivo en minúscula. Puesto que Pascal es independiente del caso, usted puede especificar los nombres de unidades en la cláusula uses en cualquier caso. Para conseguir un nombre de archivo único, el compilador Free Pascal cambia el nombre de la unidad a todo minúscula cuando está buscando por los archivos de unidad.

El compilador produce archivos en minúscula, así su unidad será encontrada, incluso si su archivo fuente tiene letras en mayúscula en él. Sólo cuando el compilador intente recompilar la unidad, no encontrará su fuente a causa de las letras en mayúscula.

3.4 Unidades, librerías y *smartlinking*

El compilador Free Pascal soporta smartlinking y la creación de librerías. Sin embargo, el comportamiento por defecto es compilar cada unidad en un archivo objeto grande, qué será enlazado como un todo en su programa.

No sólo es posible compilar una librería compartida bajo Windows 32−BIT y Linux, sino que también es posible tomar unidades existentes y ponerlas juntas en una librería estática o compartida (usando la herramienta ppumove).

3.5 Creando un ejecutable destinados para GO32V1 y PMODE/DJ

La plataforma GO32V1 no está oficialmente soportada más, así que esta sección es de interés sólo a una persona que quiera hacer binarios go32V1 de cualquier forma.

3.5.1 GO32V1

Cuando esté compilando bajo DOS, GO32V2 es el destino por defecto. Sin embargo, si usa go32V1 (usando el conmutador −TGO32V1), el proceso de la compilación le deja a usted un archivo el cual no puede ejecutar inmediatamente. Hay 2 cosas que puede hacer cuando la compilación ha terminado.

La primera cosa es usar el extensor del DOS de D.J. Delorie para ejecutar su programa:

```
go32 prog
```

Eso está bien para probar, pero si quiere usar un programa regularmente, sería más fácil si podría sólo escribir el nombre del programa, es decir.

```
prog
```

Esto puede ser logrado haciendo un ejecutable de DOS de su programa compilado.

Hay dos formas de crear un ejecutable de DOS (bajo DOS solamente):

1. Si el GO32.EXE está instalado ya en las computadoras donde el programa debe de ejecutarse, debe de copiar sólo un programa llamado STUB.EXE en el inicio del archivo AOUT. Esto se logra con el programa AOUT2EXE.EXE. El cual viene con el compilador:

   ```
   AOUT2EXE PROG
   ```

   y consigue un ejecutable de DOS el cual carga automáticamente el GO32.EXE. El ejecutable GO32.EXE debe de estar en el directorio actual o estar en un directorio en la variable PATH .

1. La segunda forma para crear un ejecutable de DOS es poner a GO32.EXE a el inicio del archivo AOUT. Para hacer esto, en el indicador de comando, escriba:

   ```
   COPY /B GO32.EXE+PROG PROG.EXE
   ```

   (asumiendo que Free Pascal creó un archivo llamado PROG, por supuesto). Este se convierte entonces en un ejecutable independiente para DOS, que no necesita el GO32.EXE en la máquina donde debe ejecutarse.

3.5.2 PMODE/DJ

Usted puede también usar el extensor PMODE/DJ para ejecutar sus aplicaciones de Free Pascal. Para hacer un ejecutable que trabaje con el extensor PMODE, puede simplemente crear un ejecutable GO32V2 (el por defecto), y entonces convertirlo a un ejecutable PMODE con los dos comandos extras siguientes:

1. Primero, elimine el encabezado de GO32V2 del ejecutable:

   ```
   EXE2COFF PROG.EXE
   ```

(suponemos que PROG.EXE es el programa generado por el proceso de la compilación). (− Falta ")"−)

2. En secundo lugar, añadir el pedazo de PMODE:

```
COPY /B PMODSTUB.EXE+PROG PROG.EXE
```

Si PMODSTUB.EXE no está en su directorio local, necesita proveerle la ruta entera.

Eso es. No se necesitan pasos adicionales para crear un extensor de PMODE ejecutable.

Esté enterado, a pesar de, que el extensor de PMODE no soporta la memoria virtual, así si está corto de memoria, puede ejecutarse con problema. También, no hay oficialmente ayuda para el extensor PMODE/DJ. Apenas sucede que el compilador y algunos de los programas que genera, se ejecutan bajo este extensor también.

3.6 Reduciendo el tamaño de su programa

Cuando creó su programa, es posible reducir su tamaño. Esto es posible, porque el compilador deja un montón de información en el programa que, estrictamente hablando, no es requerida para la ejecución de éste. El exceso de información puede ser removida con un pequeño programa llamado strip . Viene con el entorno de desarrollo GO32 bajo DOS , y es estándar en máquinas con Linux donde usted puede desarrollar. El uso es simple. Sólo escriba:

```
strip prog
```

En la línea de comando, y el programa strip removerá toda la información innecesaria de su programa. Esto puede conducir a reducciones de tamaño de hasta un 30%.

Comentario: En la versión de Win32, strip es llamado stripw .

Usted puede usar el conmutador −Xs para dejar que el compilador haga esa eliminación automáticamente al momento de compilar un programa (el conmutador no tiene efecto cuando se está compilando unidades).

Otra técnica para reducir el tamaño de un programa es usar *smartlinking* . Normalmente, las unidades (incluyendo la unidad de sistema) están enlazadas como un todo. Es sin embargo posible compilar unidades tales que puedan ser *smartlinked* . Esto quiere decir que sólo las funciones y procedimientos están enlasadas en su programa, sacando cualquier código innecesario. Esta técnica está descripta por completo en la guía de los programadores.

4 Problemas compilando

4.1 Problemas generales

- **IO−error −2 at ... :** Bajo Linux puede conseguir este mensaje en el arranque del compilador. Significa típicamente que el compilador no encuentra el archivo de las definiciones de error. Puede corregir este error bajo Linux con la opción −Fr (véase 5.1).

- **Error : File not found : xxx** o **Error: couldn't compile unit xxx:** Esto sucede típicamente cuando su ruta de la unidad no está ajustada correctamente. Recuerde que el compilador busca las unidades sólo en el directorio actual, y en el directorio donde está el compilador en sí mismo. Si usted quiere que busque también en otra parte, usted debe explícitamente decirle que hacer usando así la opción −Fu (véase 5.1). O debe configurar (− set op??? ta' raro−) un archivo de configuración.

4.2 Problemas que puede encontrar bajo DOS

- **No space in environment.**

  Un mensaje de error como éste puede ocurrir, si llama a SET_PP.BAT en el AUTOEXEC.BAT. Para resolver este problema, debe extender su memoria de entorno, Para hacerlo, busque una línea en el CONFIG.SYS como

  SHELL=C:\DOS\COMMAND.COM

  y cambiela a lo siguiente:

  SHELL=C:\DOS\COMMAND.COM /E:1024

  Puede sólo necesitar especificar un valor más alto, si este parámetro ya está ajustado.

- **Coprocessor missing**

  Si el compilador escribe un mensaje de que no hay coprocesador, instale la emulación del coprocesador.

- **Not enough DPMI memory**

  Si usted quiere usar el compilador con DPMI debe de tener al menos 7−8 MB de memoria DPMI libre, pero 16 MB es una cantidad más realista.

5 Configuración del compilador

La salida del compilador puede ser controlada de muchas formas. Se puede hacer esencialmente de dos formas distintas:

- Usando las opciones de línea de comando.
- Usando el archivo de configuración: ppc386.cfg.

El compilador lee primero el archivo de configuración. Sólo entonces las opciones de la línea de comando son revisadas. Esto crea la posibilidad para ajustar algunas opciones básicas en el archivo de configuración, y al mismo tiempo puede todavía ajustar algunas opciones específicas cuando está compilando alguna unidad o programa. Primero listamos las opciones de la línea de comando, y entonces explicamos cómo especificar las opciones de la línea de comando en el archivo de configuración. Al leer esto, tenga presente que las opciones son sensibles a las mayúsculas y minúsculas. Mientras que esto es normal para Linux, no lo es bajo DOS.

5.1 Usando las opciones de la línea de comando

La disponibilidad de las opciones para la versión 1.0.2 (− Error en la documentación: dice "0.99.10" no "1.0.2"−) del compilador están listada por categoría (véase el apéndice A para un listado como es generado por el compilador):

5.1.1 Opciones generales

−h si especifica esta opción, el compilador hace salir una lista de todas las opciones, y termina después de eso.

−? idéntico como −h, esperando después de cada pantalla completa por la entrada de una tecla.

−i Esta opción le dice al compilador que imprima la información de derechos de autor. Puede darle una opción, como:

−ixxx donde **xxx** puede ser una de la siguiente:

D : Devuelve la fecha del compilador.

V : Devuelve la versión del compilador.

SO : Devuelve el S.O. del compilador.

SP : Devuelve el procesador del compilador.

TO : Devuelve el destino del compilador.

TP : Devuelve el destino del procesador

−l Esta opción le dice al compilador que imprima el logo de Free Pascal sobre la salida estándar. También le da el número de versión de Free Pascal.

−n Dice al compilador que no lea el archivo por defecto de configuración. Usted todavía puede pasar un archivo de configuración con la opción @.

5.1.2 Opciones para conseguir realimentación

−vxxx Sea detallado. xxx es una combinación de lo siguiente:

- e : Dice al compilador que sólo muestre errores. Esta opción está activada por defecto.
- i : Dice al compilador que muestre alguna información en general.
- w : Dice al compilador que publique alertas.
- n : Dice al compilador que publique notas.
- h : Dice al compilador que publique sugerencias.
- l : Dice al compilador que muestre los números de línea mientras procesa un archivo. Los números se muestran por 100.
- u : Dice al compilador que imprima información sobre las unidades que carga.
- t : Dice al compilador que imprima los nombres de los archivos que intenta abrir.
- p : Dice al compilador que imprima los nombres de los procedimientos y funciones mientras los está procesando.
- c : Dice al compilador que le advierta cuando procesa un condicional.

- m : Dice al compilador que escriba cuales macros están definidos.

- d : Dice al compilador que escriba otra información de depuración.

- a : Dice al compilador que escriba toda la información posible (esto es lo mismo que especificar todas las opciones).

- 0 : Dice al compilador que no escriba mensajes. Esto es útil cuando usted quiere reemplazar la configuración por defecto en el archivo de configuración.

- b : Dice al compilador que muestre todas las declaraciones de procedimientos si ocurre un error de sobrecarga de función.

- x : Dice al compilador que saque alguna información del ejecutable (sólo para la plataforma Win32).

- r : Modo de compatibilidad Rhide/GCC: formatea los errores diferentemente, así son entendidos por RHIDE.

5.1.3 Opciones que conciernen a los archivos y directorios

−exxx xxx especifica el directorio donde el compilador puede buscar los ejecutables as (el ensamblador) y ld (el enlazador).

−FD igual que −e.

−Fexxx Esta opción dice al compilador que escriba errores, etc. al archivo llamado **xxx** .

−FExxx Dice al compilador que escriba el ejecutable y las unidades en el directorio **xxx** en vez del directorio actual.

−Fixxx Añade **xxx** a la ruta de búsqueda del archivo incluido.

−Flxxx Añade **xxx** a la ruta de búsqueda de la librería, y se pasa al enlazador.

−FLxxx (sólo Linux) Dice al compilador que use **xxx** como el enlazador dinámico. Por defecto este es /lib/ld−linux.so.2 , o /Hlib/ld−linux.so.1, dependiendo de cuál se encuentra uno primero.

−Foxxx Añade **xxx** a la ruta de búsqueda del archivo objeto. Esta ruta es usada cuando se busca por archivos que necesitan ser enlazados adentro.

−Frxxx **xxx** especifica el archivo que contiene los mensajes del compilador. Por defecto el compilador tiene los mensajes incluidos. Especificando esta opción reemplazará los mensajes por defecto.

−Fuxxx Añade **xxx** a la ruta de búsqueda de la unidad. Las unidades se buscan primero en el directorio actual. Si no se encuentran ahí entonces el compilador las busca en la ruta de la unidad. Usted debe proveer siempre la ruta a la unidad de sistema.

−FUxxx Dice al compilador que escriba las unidades en el directorio **xxx** en vez del directorio actual. Esto reemplaza la opción **−FE**.

−Ixxx Añade **xxx** a la ruta de búsqueda del archivo incluido. Esta opción tiene el mismo efecto que **−Fi**.

−P Usa tuberías en vez de archivos al ensamblar. Esto puede acelerar el compilador sobre OS/2 y Linux. Sólo con los ensambladores (tales como GNU as) que soportan entubado...

5.1.4 Opciones que controlan la clase de salida.

Para más información sobre estas opciones, véase también la guía del Programador.

−a Dice al compilador que no borre los archivos generados por el ensamblador (no cuando se usa el ensamblador interno). Esto también cuenta para (posiblemente) el archivo de comandos por lotes generado.

−al Dice al compilador que incluya las líneas del código fuente en el archivo del ensamblador como comentarios.

−ar Dice al compilador que liste la asignación de registro y deje información en el archivo del ensamblador. Esto es principalmente realizado para depurar el código generado por el compilador (−Error: bythe−).

−at Dice al compilador que liste información acerca de asignaciones y liberaciones temporales en el archivo del ensamblador.

−Axxx Especifica qué clase de ensamblador debe ser generado. Aquí xxx es uno de lo siguiente:

as ensamblado usando GNU as.
asaout ensamblado usando GNU as para aout (Go32v1).
nasmcoff archivo coff (Go32v2) usando Nasm.
nasmelf archivo elf32 (Linux) usando Nasm.
nasmobj archivo objeto usando Nasm.
masm archivo objeto usando Masm (Microsoft).
tasm archivo objeto usando Tasm (Borland).
coff archivo objeto coff (Go32v2) usando el escritor interno de objeto binario.

pecoff archivo objeto pecoff (Win32) usando el escritor interno de objeto binario.

-B Dice al compilador que re-compile todas las unidades usadas, incluso si los fuentes de la unidad no cambiaron desde la última compilación.

-b Dice al compilador que genere información de examinador. Esta información puede ser usada por un Entorno de Desarrollo Integrado (*IDE por sus siglas en inglés*) para dar información sobre clases, objetos, procedimientos, tipos y variables en una unidad.

-bl Es igual a `-b` pero también genera información acerca de variables locales, tipos y procedimientos.

-CD Crea una librería dinámica. Esto se usa para transformar unidades en librerías enlasables dinámicamente en LINUX.

-Chxxx Reserva xxx bytes del montón. xxx debe de estar entre 1024 y 67107840.

-Ci Genera código de chequeo de Entrada/Salida. En caso de que algún código de entrada/salida de su programa retorne un estado de error, el programa saldrá con un error de tiempo de ejecución. Cuál error se genera dependiendo del error de E/S.

-Cn Omite la etapa de enlazado.

-Co Genera código que chequea el desbordamiento de un Entero. En caso de errores de enteros, un error de tiempo de ejecución será generado por su programa.

-Cr Genera código de chequeo de Rango. En caso de que su programa accese un arreglo de elemento con un índice inválido, o si es incrementado un tipo enumerado más allá de su alcance, un error de tiempo de ejecución se generará.

-Csxxx Ajusta el tamaño de la pila a `xxx`.

-Ct Genera código de chequeo de la pila. En caso de que su programa realice un fallo de operación de la pila, un error de tiempo de ejecución será generado.

-CX Crea una unidad *smartlinked* cuando una unidad se esta escribiendo. Smartlinking enlazará solamente en las partes del código que son realmente necesitadas por el programa. Todo el código sin uso se saca. Esto puede conducir a binarios sustancialmente más pequeños.

-dxxx Define el símbolo de nombre `xxx`. Esto puede ser usado para compilar condicionalmente partes de su código.

-E Lo mismo que `-Cn`.

-g Genera información de depuración para depurar con `gdb`.

-gg Idéntico a `-g`.

-gd Genera información de depuración para `dbx`.

-gh Use la unidad heaptrc (véase la Referencia de la unidad).

-gc Genera chequeos para los punteros.

-Oxxx Optimiza la salida del compilador; `xxx` puede tener uno de los valores siguientes:

g optimice para el tamaño, trata de generar código más pequeño.
G optimice para el tiempo, trata de generar código más rápido (defecto).
r mantiene ciertas variables en los registros (experimental, use con precaución).
u optimizaciones inciertas.
1 optimizaciones de nivel 1 (optimizaciones rápidas).
2 optimizaciones de nivel 2 (`-O1` más algunas optimizaciones más lentas).
3 optimizaciones de nivel 3 (`-O2` más `-Ou`).

Pn (Sólo Intel) Procesador especificación: `n` puede ser uno de

1 optimice para 386/486
2 optimice para Pentium/Pentium MMX (tm)
3 optimice para PentiumPro/PII/Cyrix 6x86/K6 (tm)

El efecto exacto de estos efectos pueden ser encontrados en la guía del Programador.

-oxxx Dice a el compilador que use `xxx` como el nombre del archivo de salida (ejecutable). Sólo con programas.

-pg Genera código de perfilado para gprof.

-s Dice al compilador que no llame al ensamblador y enlace. En vez, el compilador escribe un archivo de comandos, PPAS.BAT bajo DOS, o ppas.sh bajo LINUX, el cual puede entonces ser ejecutado para producir un ejecutable. Esto puede ser usado para acelerar el proceso de compilado o para depurar la salida del compilador.

-Txxx Especifica el sistema operativo de destino. `xxx` puede ser uno de lo siguiente:

- **GO32V1** : DOS y versión 1 del extensor DJ DELORIE (no más mantenido).
- **GO32V2** : DOS y versión 2 del extensor DJ DELORIE.
- **LINUX** : LINUX.
- **OS2** : OS/2 (2.x) usando el extensor `EMX`.
- **WIN32** : WINDOWS 32 bit.

-uxxx Indefine el símbolo xxx. Esto es lo opuesto a la opción `-d`.

-Xx Opciones de los ejecutables. Esto dice al compilador qué clase de ejecutable se debe de generar. El parámetro `x` puede ser uno de lo siguiente:

- **c** : (sólo Linux) Enlace con la librería de C. Usted debe solamente usar esto cuando usted inicia un porte de Free Pascal a otro sistema operativo.
- **D** : Enlace con las librerías dinámicas (define el símbolo `FPC_LINK_DYNAMIC`).
- **s** : Elimina los símbolos del ejecutable.
- **S** : Enlace con las unidades estáticas (define el símbolo `FPC_LINK_STATIC`).
- **X** : Enlace con las unidades *smartlinked* (define el símbolo `FPC_LINK_SMART`).

5.1.5 Opciones referentes a las fuentes (opciones del lenguaje).

Para más información sobre estas opciones, véase también la guía del Programador.

−Rxxx Especifica que clase de ensamblado usted usa en sus bloques de ensamblador `asm` . Aquí `xxx` es uno de lo siguiente:

> **att** bloques `asm` conteniendo ensamblado estilo `AT&T`. Este es el estilo por defecto.
> **intel** bloques asm conteniendo ensamblado estilo `Intel`.
> **direct** bloques `asm` que deben ser copiados como están en el ensamblador, sólo remplazando algunas variables. (*−algo de más: file−*)

−S2 Activa las extensiones de Delphi 2. Este es diferente a `−Sd` porque algunos constructores de Free Pascal están todavía disponible para usted.

−Sc Soporte operadores de estilo `C`, ej.: `*=, +=, /= and −=` .

−Sd Dice al compilador que sea compatible con Delphi. Este es más estricto que la opción `−S2`, ya que algunas extensiones de `FPC` son desactivadas.

−SeN El compilador se detiene después de `N` errores. Normalmente, el compilador trata de continuar compilando después de un error, hasta que se alcancen 50 errores, o se produzca un error fatal, y entonces se detiene. Con este interruptor, el compilador parará después de `N` error (si `N` se omite, se asume un valor de 1 por defecto).

−Sg Soporte los comandos `label` y `goto`. Por defecto éstos no son soportados. Usted debe también especificar esta opción si usa etiquetas en declaraciones de ensamblador (si usa el ensamblado estilo `AT&T`).

−Sh Use `ansistrings` por defecto para las cadenas. Si esta palabra clave se especifica, el compilador interpretará la palabra clave `string` como una `ansistring` . De otro modo se supone que sea una cadena corta (estilo TP).

−Si Soporte `INLINE` al estilo de `C++`.

−Sm Soporte macros al estilo `C`.

−So Trata de ser compatible con Borland TP 7.0 (sin sobre carga de función, etc.).

−Sp Trata de ser compatible con `gpc` (compilador GNU Pascal).

−Ss El nombre de los constructores debe de ser `init`, y el nombre de los destructores debe de ser `done`.

−St Permite la palabra clave `static` en los objetos.

−Un No chequee el nombre de la unidad. Normalmente, el nombre de la unidad es el mismo como el del nombre de archivo. Esta opción permite que ambos sean diferentes.

−Us Compile una unidad de sistema. Esta opción causa que el compilador defina sólo algunos tipos muy básicos.

5.2 Usando el archivo de la configuración.

Usando el archivo de configuración ppc386.cfg es una alternativa a las opciones de la línea de comando. Cuando se encuentra un archivo de configuración, se lee, y las líneas en éste son tratadas como usted las escribe en la línea de comando.

Se tratan antes que las opciones que usted escribe en la línea de comando.

Usted puede especificar comentarios en el archivo de la configuración con el signo #. Cualquier cosa después del # será ignorado.

El algoritmo para determinar cual archivo es usado como un archivo de configuración se describe en 3.1 en la página 17.

Cuando el compilador ha acabado de leer el archivo de la configuración, continúa tratando las opciones de la línea de comando.

Una opción de la línea de comando le permite a usted especificar un segundo archivo de configuración: especificando @foo en la línea de comando abrirá el archivo foo, y lee opciones posteriores desde allí. Cuando el compilador ha acabado de leer este archivo, continúa procesando la línea de comando.

El archivo de la configuración permite una cierta clase de preprocesado. Entiende las directivas siguientes, que usted debe poner en la primera columna de una línea:

#IFDEF

#IFNDEF

#ELSE

#ENDIF

#DEFINE

#UNDEF

#WRITE

#INCLUDE

#SECTION

Ello trabaja de la misma manera como su contraparte `{$...}` en Pascal.

Lo que sigue es una descripción de las diferentes directivas.

5.2.1 #IFDEF

Sintaxis:

```
#IFDEF nombre
```

Las líneas que siguen a `#IFDEF` se dejan de leer si el nombre de la palabra clave que sigue no está definida.

Se leen hasta que las palabras claves `#ELSE` o `#ENDIF` son encontradas, después de lo cual se reasume el procesado normal.

Ejemplo:

```
#IFDEF VER0_99_5
-Fu/usr/lib/fpc/0.99.5/linuxunits
#ENDIF
```

En el ejemplo anterior, `/usr/lib/fpc/0.99.5/linuxunits` será añadido a la ruta si usted está compilando con la versión 0.99.5 del compilador.

5.2.2 #IFNDEF

Sintaxis:

```
#IFNDEF nombre
```

Las líneas que siguen a `#IFNDEF`*(−Error: dice #IFDEF−)* se dejan de leer si el nombre de la palabra clave que sigue está definida.

Se leen hasta que las palabras claves `#ELSE` o `#ENDIF` son encontradas, después de lo cual se reasume el procesado normal.

Ejemplo:

```
#IFNDEF VER0_99_5
-Fu/usr/lib/fpc/0.99.6/linuxunits
#ENDIF
```

En el ejemplo anterior, `/usr/lib/fpc/0.99.6/linuxunits` será añadido a la ruta si usted NO está compilando con la versión 0.99.5 del compilador.

5.2.3 #ELSE

Sintaxis:

```
#ELSE
```

`#ELSE` puede ser especificado después de una directiva `#IFDEF` o `#IFNDEF` como una alternativa. Las líneas siguientes a `#ELSE` se dejan de leer si el precedido `#IFDEF` o `#IFNDEF` fue aceptado.

Se saltan hasta que la palabra clave `#ENDIF` se encuentra, después de lo cual se reasume el procesado normal.

Ejemplo: *(− Este ejemplo tiene un error −)*

```
#IFDEF VER0_99_5
−Fu/usr/lib/fpc/0.99.6/linuxunits
#ELSE
−Fu/usr/lib/fpc/0.99.5/linuxunits
#ENDIF
```

En el ejemplo anterior, `/usr/lib/fpc/0.99.5/linuxunits` será añadido a la ruta si usted está compilando con la versión `0.99.5` del compilador, si no `/usr/lib/fpc/0.99.6/linuxunits` será añadido a la ruta.

5.2.4 #ENDIF

Sintaxis:

```
#ENDIF
```

`#ENDIF` marca el fin de un bloque que inició con `#IF(N)DEF` , posiblemente con un `#ELSE` en medio.

5.2.5 #DEFINE

Sintaxis:

```
#DEFINE nombre
```

`#DEFINE` define una palabra clave nueva. Esto tiene el mismo efecto que la opción `−dname` de la línea de comando.

5.2.6 #UNDEF

Sintaxis:

```
#UNDEF nombre
```

#UNDEF indefine una palabra clave si existió. Esto tiene el mismo efecto que la opción -uname de la línea de comando.

5.2.7 #WRITE

Sintaxis:

```
#WRITE Mensaje de Texto
```

#WRITE escribe Mensaje de Texto a la pantalla. Esto puede ser útil para presentar advertencias si algunas opciones se seleccionan.

Ejemplo:

```
#IFDEF DEBUG
#WRITE Seleccionando depurado ON...
-g
#ENDIF
```

Si DEBUG está definido, ésto producirá una línea

```
Seleccionando depurado ON...
```

y entonces encenderá el depurado de información en el compilador.

5.2.8 #INCLUDE

Sintaxis:

```
#INCLUDE nombrearchivo
```

#INCLUDE instruye al compilador a que lea los contenidos de nombrearchivo antes de continuar procesando las opciones en el archivo actual.

Esto puede ser útil si usted quiere tener un archivo de configuración particular para un proyecto (o, bajo LINUX, en su directorio HOME), pero todavía quiere tener las opciones globales que están definidas en un archivo global de configuración.

Ejemplo:

```
#IFDEF LINUX
  #INCLUDE /etc/ppc386.cfg
#ELSE
  #IFDEF GO32V2
    #INCLUDE c:\pp\bin\ppc386.cfg
  #ENDIF
#ENDIF
```

Esto incluirá `/etc/ppc386.cfg` si usted está en una máquina Linux, e incluirá `c:\pp\bin\ppc386.cfg` en una máquina del DOS.

5.2.9 #SECTION

Sintaxis:

```
#SECTION nombre
```

La directiva `#SECTION` actúa como una directiva `#IFDEF` , sólo que no requiere una directiva `#ENDIF`.

El nombre especial `COMMON` siempre existe, o sea las líneas que siguen a `#SECTIONCOMMON` siempre se leen.

5.3 Sustitución de variable en las rutas.

Para evitar de tener que editar sus archivos de la configuración demasiado a menudo, el compilador permite que usted especifique las variables siguientes en las rutas que usted suministra al compilador:

FPCVER se reemplaza por la cadena de la versión completa del compilador.

FPCDATE se reemplaza por la fecha del compilador.

FPCTARGET se reemplaza por el CPU de destino del compilador (desaprobado).

FPCCPU se reemplaza también por el CPU de destino del compilador.

TARGET se reemplaza por el S.O. de destino (desaprobado).

FPCOS se reemplaza por el S.O. de destino.

Para tener estas variables substituibles, sólo insértela con un $ de prefijo, como sigue:

```
-Fu/usr/lib/fpc/$FPCVER/rtl/$FPCOS
```

Esto es equivalente a

```
-Fu/usr/lib/fpc/0.99.12a/rtl/linux
```

Si la versión del compilador es 0.99.12a y el S.O. de destino es Linux.

Estos reemplazos son válidos en la línea de comandos y también en el archivo de la configuración.

En la línea de comandos de Linux, usted debe de tener cuidado de escapar el `$` ya que de otra manera la `shell` ampliará la variable para usted, lo cual puede tener efectos indeseados.

6 El IDE

El IDE (**I**ntegrated **D**evelopment **E**nvironment, Entorno de Desarrollo Integrado) provee una interfaz de usuario confortable para compilar. Contiene un editor con realce de sintaxis, un depurador, visualizador de símbolo, etc. (− Falta , −) El IDE es una aplicación en modo de texto la cual tiene el mismo aspecto y sentir sobre todos los sistemas operativos soportados. Está modelado en el IDE de Turbo Pascal, así que muchas personas deberían de sentirse cómodo usándolo.

Actualmente, el IDE está disponible para el DOS, Windows y Linux.

6.1 Primeros pasos con el IDE

6.1.1 Iniciando el IDE

El IDE es iniciado entrando el comando:

```
fp
```

en la línea de comandos. También puede ser iniciado desde una interfaz de usuario gráfica tal como Windows. **Comentario:** Bajo Windows, es posible cambiar entre modo de ventana y pantalla completa presionando ALT−ENTER. (− Un) de más−)

6.1.2 Opciones en la línea de comandos del IDE

Cuando estés iniciando el IDE, se pueden pasar opciones en la línea de comandos:

```
fp [−option] [−option] ... <file name> ...
```

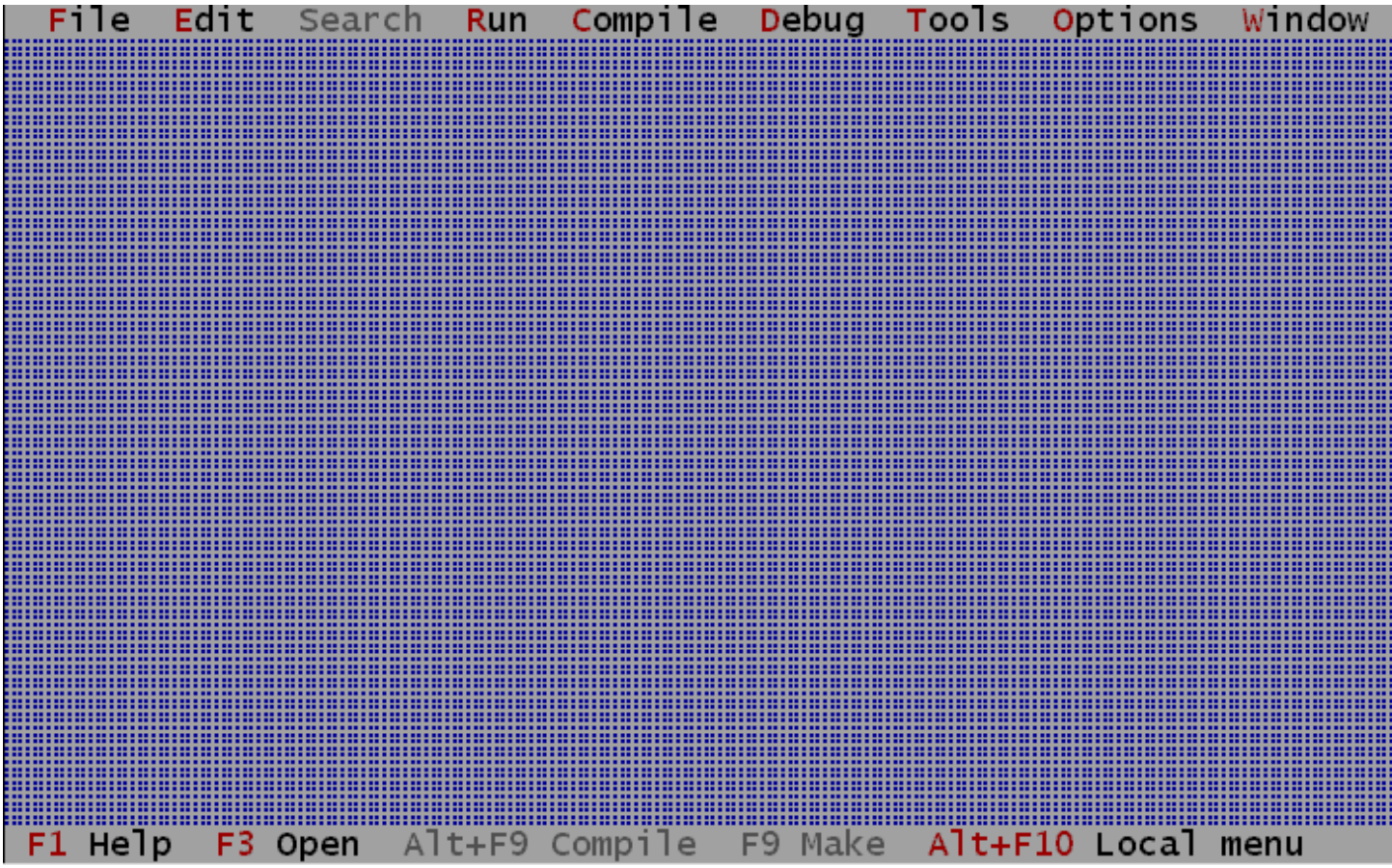
`option` es una de las opciones siguientes (las letras de las opciones son insensibles a las mayúsculas y minúsculas):

−N (DOS solamente) No use nombres de archivos largos. Windows 95 y posteriores versiones de Windows proveen una interfaz a aplicaciones del DOS para accesar nombres de archivos largos. El IDE usa esta interfaz por defecto para accesar archivos. Bajo algunas circunstancias, esto puede ser un problema. Esta opción le dice al IDE que no use nombres de archivos largos.

−Cfilename Esta opción, seguida por un nombre de archivo, le dice a el IDE que lea las opciones desde `filename`. Ahí no debería de haber espacio en blanco entre el nombre del archivo y el `−C`.

−F Use caracteres gráficos alternativos. Esto puede ser usado para correr el IDE sobre Linux en una Terminal−X o a través de una sección Telnet.

Figura 6.1: La pantalla del IDE inmediatamente después de iniciado



−R Después de iniciado el IDE, éste cambia automáticamente al directorio del cual estaba activo cuando el IDE salió la última vez.

−S Inhabilita el ratón. Cuando esta opción se usa, el ratón es inhabilitado, incluso si está presente un ratón.

Los archivos que se dan en la línea de comandos son cargados en la ventana del editor automáticamente.
Comentario: Bajo DOS/Win32, el primer carácter de una opción de línea de comandos puede ser un carácter `/` en vez de un carácter `−`. Así que `/S` es equivalente a `−S` .

6.1.3 La pantalla del IDE

Después de iniciado, la pantalla del IDE puede verse como la figura (6.1). En la parte superior de la pantalla la *barra de menú* es visible, en la parte inferior la *barra de estado*. El espacio vacío entre ellas se llama el *escritorio*.

La barra de estado muestra los atajos del teclado para comandos usados frecuentemente y permite un rápido acceso a esos comandos clicleando con el ratón. En el borde derecho de la barra de estado, el monto actual de memoria sin usar es presentado. Esto es sólo una indicación, ya que el IDE intenta alojar más memoria del sistema operativo si corre sin memoria.

El menú provee acceso a todas las funcionalidades del IDE y en el borde derecho del menú, se presenta un reloj.

Se puede salir del IDE seleccionando **"File|Exit"** del menú [1] o presionando ALT−X.(− *Revisar línea* −)
Comentario: Si un archivo fp.ans se encuentra en el directorio actual, entonces es cargado y usado para pintar el fondo.
Ese archivo debería contener comandos de dibujos ANSI para dibujar sobre la pantalla.

6.2 Navegando en el IDE

(− *Falta ":"*−)
El IDE puede ser navegado con ambos: el teclado y un ratón, si el sistema está equipado con un ratón.

[1] **"File|Exit"** quiere decir seleccionar el ítem "Exit" en el menú "File".

6.2.1 Usando el teclado

Todas las funcionalidades del IDE están disponibles a través del uso del teclado.

- Es usado para escribir y navegar a través de las fuentes.
- Editando comandos tales como copiando y pegando texto.
- Moviendo y cambiando el tamaño de las ventanas.
- Puede ser usado para accesar el menú, presionando ALT y la letra realzada apropiada del menú, o presionando F10 y navegando a través del menú con las teclas de flechas.

  Más información sobre el menú puede ser encontrada en la sección 6.4, página 37.

- Muchos comandos en el IDE están ligados a atajos, es decir escribiendo una combinación especial de teclas se ejecutará un comando inmediatamente.

  Comentarios:

- Cuando se está trabajando en una Terminal−X de Linux o a través de una sección de Telnet, la combinación de teclas con ALT puede no estar disponible. Para remediar esto, la combinación CTRL−Z puede ser presionada primero. Esto quiere decir que por ejemplo ALT−X puede ser remplazada por CTRL−Z X.(−Cuál es el comentario?−)

- Una referencia completa de todos los atajos del teclado pueden ser encontrados en la sección 6.14, página 79.

6.2.2 Usando el ratón

Si el sistema está equipado con un ratón, se puede usar para trabajar con el IDE. El botón izquierdo es usado para seleccionar ítems del menú, presionar botones, seleccionar bloques de texto, etc. (− Falta ,−)

El botón derecho del ratón es usado para accesar el menú local, si está disponible. Presionando la tecla CTRL o ALT y haciendo clic con el botón derecho se ejecutarán funciones definidas por el usuario, véase la sección 6.12, página 75.

Comentarios:

1. Ocasionalmente, el manual usa el término "arrastrar el ratón". Esto quiere decir que el ratón es movido mientras el botón izquierdo del ratón está presionado.
2. La acción de los botones del ratón se pueden invertir, es decir las acciones del botón izquierdo del ratón pueden ser asignadas al botón derecho del ratón y viceversa [2] . Por todo el manual, se asume que las acciones de los botones del ratón no están invertidas.
3. El ratón no está disponible siempre, incluso si un ratón está instalado:

- El IDE está corriendo bajo Linux a través de una conexión Telnet desde una máquina Windows.
- El IDE está corriendo bajo Linux en una Terminal−X bajo X−Windows. (−No es X−Window?−)

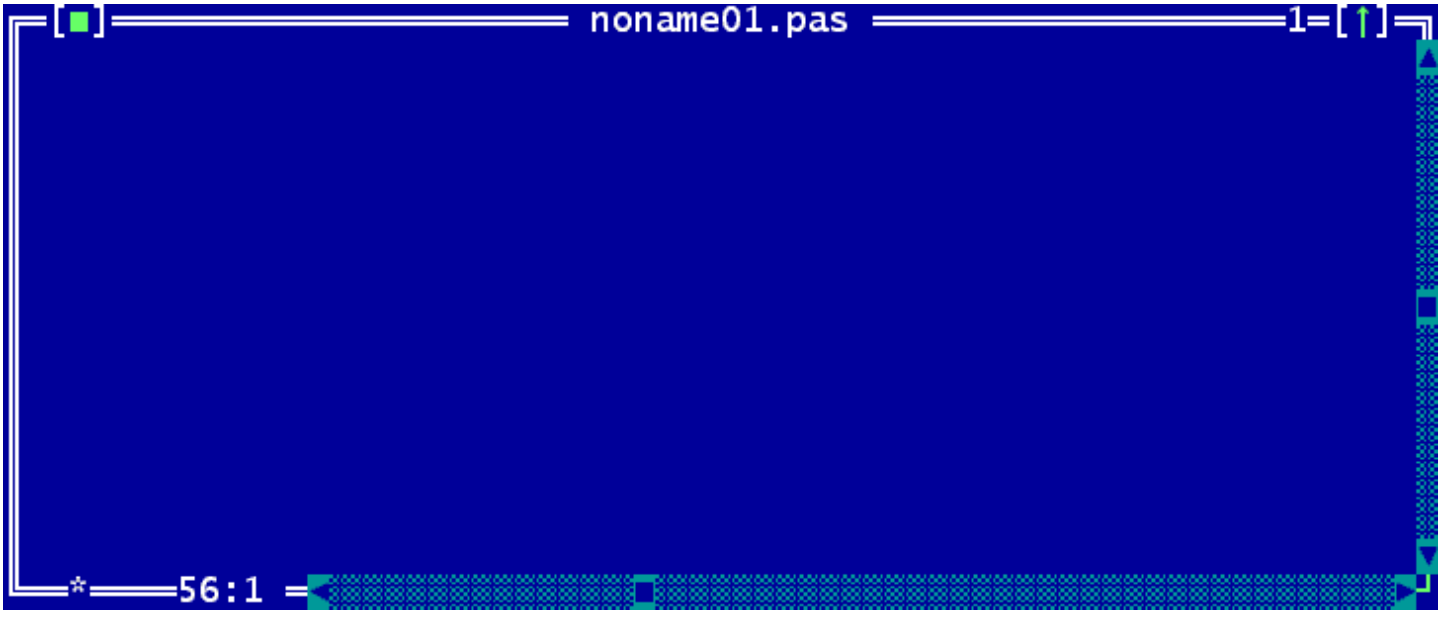
6.2.3 Navegando en los diálogos

Los diálogos usualmente tienen muchos elementos en ellos tal como botones, campos de edición, campos de memo, cuadros de lista y otros. Para activar uno de estos campos, es suficiente con:

1. Clic sobre el elemento con el ratón.
2. Presione la tecla TAB hasta que el foco alcance al ratón.

[2] Véase la sección 6.12, página 75 para más información sobre como revertir las acciones de los botones del ratón.

Figura 6.2: Una ventana común del IDE



3. Presione la letra realzada en el elemento de la etiqueta. Si el foco está actualmente sobre un elemento que permite edición, entonces ALT debería ser presionada simultáneamente con la letra realzada. Para un botón, la acción asociada con el botón entonces se ejecutará.

Dentro de los campos de edición, cuadros de lista, memos, la navegación se lleva a cabo con los comandos usuales de las teclas de flecha.