Irie Pascal User's Manual (Solaris/Sparc Edition)

Author: Stuart King

Version: Irie Pascal Version 2.6

Page: 1

TABLE OF CONTENTS

1.1 Getting help

A variety of resources are available to help you get the most out of Irie Pascal. For more information see the list below:

1.2 Installing and uninstalling

Minimum system requirements for Irie Pascal (Solaris/Sparc Edition)

- Solaris/Sparc 8 or later.
- 5 MB disk space.
- HTML or PDF viewer/browser to access the documentation.

Installing Irie Pascal (Solaris/Sparc Edition)

Irie Pascal is distributed as a compressed archive (i.e. **ips-eval.sparc.tar.Z** or **ips-260.sparc.tar.Z**).

The suggested installation procedure is as follows:

A. Log in to Solaris/Sparc as **root** or become **root** using **su**.
B. Create a directory to store Irie Pascal (e.g. **/usr/local/irie**).
C. Copy the archive into this directory.
D. Uncompress the archive, using:

```
uncompress ips-eval.sparc.tar.Z
```

or

```
uncompress ips-260.sparc.tar.Z
```

E. Extract the files, using:

```
tar xf ips-eval.sparc.tar
```

or

```
tar xf ips-260.sparc.tar
```

.
F. Either add the directory you created previously to the path or copy the executables (i.e. ipc, ivm or ivmdb) into one of the directories (e.g. **/usr/local/bin**) in the path.
G. If you have MySQL installed and you want to use it with Irie Pascal then make sure that the directory containing the MySQL dynamic client libraries (i.e. **libmysqlclient.so**) has been added to the **LD_LIBRARY_PATH** environment variable.

Uninstalling Irie Pascal (Solaris/Sparc Edition)

Irie Pascal (Solaris/Sparc Edition) does not make any behind the scenes modifications to your system or install any files on your hard disk, so uninstalling is very simple, just erase the files.

1.3 Getting started creating programs

See creating new programs for information on how to create programs.

1.4.1 Getting started with the sample programs

In order to help get you started, a number of sample programs have been included with Irie Pascal.

Some of these sample projects are listed below:

- The hello world sample program
- The hello world sample program (CGI version)
- The hello world sample program (Windows API version)

1.4.2 The hello world sample program

One of the sample programs included with Irie Pascal is named *hello*. When this program is run, it displays the message "Hello world!" on the console. Compiling and running this program is a useful way to check whether Irie Pascal is installed correctly.

1.4.3 The hello world sample program (CGI version)

One of the sample programs included with Irie Pascal is named **hellocgi**, and is a Common Gateway Interface (CGI) version of the hello sample project. When the **hellocgi** program is run, it generates a web page with the message "Hello world!". Compiling this program and installing the executable on your web server is a useful way to check whether you have correctly configured the web server to execute Irie Pascal CGI programs.

NOTE: CGI is a very simple and popular way of integrating programs with web servers. You can find further information about CGI on the web (including on the Irie Tools website at the following URL: www.irietools.com/cgi). There are also numerous books available that cover CGI.

1.4.4 The hello world sample program (Windows API version)

One of the sample programs included with Irie Pascal is named **hellowin**, and is a Windows API version of the **hello** program. When the **hellowin** program is run, it uses the Windows API to display the message "Hello world!" in a Windows message box. **NOTE:** Because this program uses the Windows API directly, it can only be run in Windows (or possibly in a Window emulator running on some other platform).

2.1 How to get help

See getting help for information on the available Irie Pascal help resources and how to access them.

2.2 How to buy

See buying licenses for information on how to buy Irie Pascal licenses.

2.3 How to contact Irie Tools

Contact Information

E-Mail:

- sales@irietools.com (for sales related enquiries)

- support@irietools.com (for customer support enquiries)
- sking@irietools.com (for all other enquiries)

Fax:

1-876-946-2703

Postal Mail:

```
Attn: Stuart King
Irie Tools
221 S. State Rd, 7, #247
Plantation, FL 33317, USA
```

2.4.1 Creating new programs

Creating the program text

You need to use a text editor (not included with the Solaris/Sparc Edition of Irie Pascal) to create the text for your Irie Pascal programs. NOTE: Solaris/Sparc usually includes at least two text editors (i.e. **vi** and **emacs**) that you can use with Irie Pascal.

Compiling your programs

You use the command-line compiler (i.e. ipc) to compile the text of your Irie Pascal programs into executable files (see using the command-line compiler for more information).

Running your programs

You use the command-line interpreter (i.e. ivm or ivmdb) to run the executable files created by the compiler (see using the interpreter for more information).

2.4.2 Opening existing programs

Editing the program text

You need to use a text editor (not included with the Solaris/Sparc Edition of Irie Pascal) to edit the text for your Irie Pascal programs. NOTE: Solaris/Sparc usually includes at least two text editors (i.e. **vi** and **emacs**) that you can use with Irie Pascal.

Compiling your programs

You use the command-line compiler (i.e. ipc) to compile the text of your Irie Pascal programs into executable files (see using the command-line compiler for more information).

Running your programs

You use the command-line interpreter (i.e. ivm or ivmdb) to run the executable files created by the compiler (see using the interpreter for more information).

2.4.3 Compiling programs

Compiling your programs

You use the command-line compiler (i.e. ipc) to compile the text of your Irie Pascal programs into executable files (see using the command-line compiler for more information).

2.4.4 Running programs

Running your programs

You use the command-line interpreter (i.e. ivm or ivmdb) to run the executable files created by the compiler (see using the interpreter for more information).

2.5.1.1 Connecting to ODBC databases

What is ODBC?

The Open Database Connectivity (ODBC) interface is widely used to access Database Management Systems (DBMSs). ODBC was originally created by Microsoft but has now become an industry standard, and is supported by all major DBMSs for Windows.

Connecting to ODBC databases

In order to connect to an ODBC database, you need to:

1. Declare a variable of the built-in type *connection*
2. Use *new* on the variable to create a *connection* object
3. Invoke the *open* method of the *connection* object

The *open* method of the *connection* object takes one argument, the connection string. ODBC connection strings can take one of two forms:

odbc-connection-string = odbc-connection-string-1 | odbc-connection-string-2

odbc-connection-string-1 = 'ODBC' ';' dsn-parm ';' uid-parm ';' pwd-parm [';' [driver-specific-text]]

odbc-connection-string-2 = 'ODBC' ';' name ';' [uid] ';' [password]

dsn-parm = 'DSN' '=' name

uid-parm = 'UID' '=' uid

pwd-parm = 'PWD' '=' [password]

where [] indicates optional parameters

and **name** is a valid data source name (DSN)

and **uid** is an identifier for the user accessing the database

and **password** is the password of the user accessing the database

For example DSN=test;UID=sa;PWD=

When the connection string is in the first form then it is passed, without further processing, to *SQLDriverConnect* to open the connection. When the connection string is in the second form then the **name**, **id**, and **password** parameters are extracted from the connection string, if present, and passed to *SQLConnect* to open the connection. NOTE: The first form of the connection string is the recommended form, support for the second form is provided for completeness only.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information.

The authoritative source of information about ODBC is the Microsoft Developers Network (MSDN). You can access the MSDN on the Microsoft website at msdn.microsoft.com.

2.5.1.2 Connecting to MySQL databases

What is MySQL?

MySQL is a popular open source database management system (DBMS). MySQL is particular popular for computers running Linux and FreeBSD. You can go to www.MySQL.com for more information about MySQL.

Connecting to MySQL databases

In order to connect to a MySQL database, you need to:

1. Declare a variable of the built-in type *connection*
2. Use *new* on the variable to create a *connection* object
3. Invoke the *open* method of the *connection* object

The *open* method of the *connection* object takes one argument, the connection string. MySQL connection strings take the following form:

mysql-connection-string = 'MYSQL' ';' mysql-parameter-list

mysql-parameter-list = mysql-parameter ';' mysql-parameter-list | empty

mysql-parameter = mysql-host-parameter | mysql-user-parameter | mysql-password-parameter | mysql-database-parameter | mysql-port-parameter | mysql-socket-parameter | mysql-compress-parameter

mysql-host-parameter = 'host' '=' "" host-name ""

mysql-user-parameter = 'user' '=' "" user-name ""

mysql-password-parameter = 'password' '=' "" password ""

mysql-database-parameter = 'database' '=' "" database-name ""

mysql-port-parameter = 'port' '=' port-number

mysql-socket-parameter = 'socket' '=' "" socket ""

mysql-compress-parameter = 'compress' '=' boolean-value

boolean-value = 'yes' | 'no' | 'true' | 'false'

For example MYSQL;user="testuser";database="testdb";socket="/tmp/mysql.soc";

The connection parameters are extracted from the connection string and passed to *mysql_real_connect* to open the connection. NOTE: *mysql_real_connect* is the MySQL C API function that is used to open a connection to a MySQL database.

The effect of each of the parameters is described below:

The **mysql-host-parameter** specifies the hostname or IP address of the MySQL database server. If **mysql-host-parameter** is not specified or if **host-name** is an empty string or is equal to "local-host" then the connection is opened to the local MySQL server over a UNIX socket.

The **mysql-user-parameter** specifies the username used to connect to the MySQL database server. If the **mysql-user-parameter** is not specified or if **user-name** is an empty string then the login name of the person running the application is used.

The **mysql-password-parameter** specifies the password of the user who will be connected to the database server. If the **mysql-password-parameter** is not specified or if **password** is an empty string then the connection is rejected if the user actually has a password.

The **mysql-database-parameter** specifies the initial database selected when the connection is opened. If the **mysql-database-parameter** is not specified or if **database** is an empty string then no initial database is selected. In which case you must call the **selectdatabase** method later on to select a database.

The **mysql-port-parameter** specifies the port used to remotely connect to a MySQL database server over TCP. If the **mysql-port-parameter** is not specified or if **port-number** is 0 then the default port is used.

The **mysql-socket-parameter** specifies the filename of the UNIX socket used to connect to a MySQL database server on the local machine. If the **mysql-socket-parameter** is not specified or if **socket** is an empty string then the default socket is used.

The **mysql-compress-parameter** specifies the compression is to be used when communicating with the MySQL database server. If the **mysql-compress-parameter** is not specified then compression is not used.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information.

2.5.2.1 Using the execute method

Executing Database Commands

Most major Database Management Systems (DBMSs) use the Structured Query Language (SQL) as their query language. So executing a database command usually means sending an SQL statement to the DBMS for execution. You can call the *execute* method of a *connection* object to execute a database command. Before calling the *execute* method the *connection* object must represent an open connection to a database. See connecting to ODBC databases and connecting to MySQL databases for more information about connecting to databases.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information.

2.5.3.1 Using the recordset object

Most major Database Management Systems (DBMSs) use the Structured Query Language (SQL) as their query language. Some SQL statements return result sets. The *recordset* object is used to send SQL statements to DBMSs and access the result set(s) returned. The *open* method of a *recordset* object is used to send the SQL statement to the DBMS and prepare to access the result set returned. After opening the *recordset* object you can access the result set, one record at a time.

The *field* property of a *recordset* object is used to retrieve the value of a field of the current record in the result set. For example if you have a *recordset* object **rs** with an open recordset and the records in the recordset have a field named **last_name** then

```
rs.field('last_name')
```

contains the contents of the field **last_name** for the current record in the recordset.

field is a read-only property of type *variant*.

NOTE: The *field* property is read-only so you can not use it to change the contents of the record fields (i.e. you can't use this property to effect the data in the database). If you want to affect the data in the database you should call the *execute* method of an *connection* object and pass it a SQL UPDATE statement.

NOTE: Since the *field* property is used so often Irie Pascal allows you to leave out the name of the property. So for example

```
rs.('last_name')
```

is equivalent to

```
rs.field('last_name')
```

Other useful methods and properties of *recordet* objects are:

- The *movenext* method of a *recordset* object can be used to advance to the next record in the result set returned by a query.
- The *eof* property of a *recordset* object indicates whether or not the current record of the result set is the last record.
- The *close* method of a *recordset* object should be used when you have finished accessing a result

set.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information.

2.6.1 Programing Sockets

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information on how to program sockets.

2.7.1 Distributing Irie Pascal programs

Distributing Irie Pascal Programs

So you have written your program, finished testing it, and now you are ready to distribute it. You probably can't depend on the users of your program having Irie Pascal installed on their computers, so you will need to distribute the IVM Interpreter (i.e. ivm or ivmdb) in addition to the executable file containing your program. If you have purchased a license to use Irie Pascal, then you may have the right to distribute the files necessary to run your programs (see the license agreement for details).

3.1.1 Using the command-line compiler

The Irie Pascal command-line Pascal compiler translates Pascal source programs into Irie Virtual Machine executables (**.IVM** executables), or (**.EXE** executables).

Command-line Compiler (ipc) Syntax

The syntax for the command-line compiler is as follow:

- **ipc** - The compiler displays a brief help screen showing the proper syntax.
- **ipc ?** - The compiler displays a more detailed help screen listing all the options you can use.
- **ipc [options] filename** - The compiler attempts to compile the file specified by **filename**, using the options specified. NOTE: **filename** can include path information, and **[x]** indicates that **x** is optional.

For example to compile the sample program **hello.pas** enter

```
ipc hello
```

or

```
ipc hello.pas
```

(assuming of course that **hello.pas** is in the current directory).

The compiler will generate a file called **hello.ivm** which contains an IVM executable. You can run it by entering

```
ivm ./hello
```

or

```
ivm ./hello.ivm
```

3.1.2.1 Compiler options overview

Compiler options are instructions to the compiler to somehow modify its behavior (usually they are instructions to enable or disable a particular compiler feature). Compiler options are entered on the command-line when you invoke the command-line compiler. **NOTE**: Compile options are case-sensitive so for example **i** and **I** are different compiler options.

There are two kinds of compiler options

1. Flag options
2. Value options

Flag Options

Flag options are used to enable or disable a compiler feature.

To enable the feature use

```
-optionor   -option+
```

where **option** is the particular compiler option.

To disable the feature use

```
-option-
```

For example the nc option is used to enable/disable the processing of nested comments. So **-nc** or **-nc+** is used to enable the processing of nested comments and **-nc-** is used to disable the processing of nested comments.

Value options are used to specify the value of some quantity.

To specify a value option use

```
-optionVALUE
```

where **option** is the particular compiler option and **VALUE** is the value being specified.

For example the mw option is used to specify the maximum number of warnings that the compiler should process. So **-mw2** is used to set the maximum number of warnings to 2.

More than one option can be specified so for example, if you want to compile the program **bad.pas** using brief messages and nested comments then you can enter

```
ipc -b -nc bad.pas
```

Options can be combined so you can also enter

```
ipc -bnc bad
```

or

```
ipc -ncb bad
```

To turn off nested comment processing and brief message then you can enter

```
ipc -nc- -b- bad
```

or

```
ipc -ncb- bad
```

Options List

The command-line compiler options are listed below:

- -s Strict var strings
- -sc Use short-circuit evaluation
- -so Maximum stack overflow checking
- -Snn Set stack size in K
- -u Trap use of undefined values
- -v Trap use of inactive variants
- -W Control warning messages

3.1.2.2.1 -aN Align on N bytes

This command-line compiler option specifies the maximum alignment used by the compiler.

```
Syntax:  -aN(Sets maximum alignment to N)
Default: Maximum alignment is 4.
Notes: See also specify align size
```

3.1.2.2.2 -ao* Trap assignment overflow errors

This command-line compiler option enables/disables assignment overflow checking.

```
Syntax:  -ao[+|-]
Default: Enabled
Notes: See also Assignment overflow checking.
```

3.1.2.2.3 -A* Enable Asserts

This command-line compiler option enables/disables asserts.

```
Syntax: -A[+|-]
Default: Enabled
Notes: See also enable asserts.
```

3.1.2.2.4 -b Use brief messages

This command-line compiler option enables/disables the brief format for messages.

```
Syntax:  -b[+|-]
Default: Disabled
Notes:
```

The verbose format for Fatal error and Error messages (which is used by default) is

Error #nn: "name" (Line l, Col c): text

where

- **nn** is a number identifying the message
- **name** is the file where the problem as detected
- **l** is the line where the problem was detected
- **c** is the column where the problem was detected

- **text** is the text of the message

The verbose format for warning messages is similar except that **Warning** is used instead of **Error**.

The brief format for Fatal error and Error messages is:

Enn: "name" l:text

where

- **nn** is a number identifying the message
- **name** is the file where the problem as detected
- **l** is the line number of the line where the problem was detected
- **text** is the text of the message

The brief format for warning messages are similar except that **W** is used instead of **E**.

3.1.2.2.5 -C Identifiers are case-sensitive

This command-line compiler option enables/disables case-sensitive identifiers.

```
Syntax:  -C[+|-]
Default: Disabled
Notes: See also make identifiers case-sensitive.
```

3.1.2.2.6 -cm20 Compatibility mode

This command-line compiler option controls whether the compiler is in version 2.0 compatibility mode.

```
Syntax:  -cm20[+|-]
Default: Disabled
Notes: See also Compatibility mode (with version 2.0).
```

3.1.2.2.7 -ead* Auto-declare input & output

This command-line compiler option enables/disables auto-declaration of input and output.

```
Syntax:  -ead[+|-]
Default: Enabled
Notes: See also auto-declare input/output.
```

3.1.2.2.8 -ebh* Allow binary & hex integers

This command-line compiler option enables/disables binary and hexadecimal integer constants.

```
Syntax:  -ebh[+|-]
Default: Enabled
Notes: See also allow binary/hexadecimal constants.
```

3.1.2.2.9 -eco* Enable non-standard constants

This command-line compiler option enables/disables non-standard constants.

Syntax: **-eco[+|-]**
Default: Enabled
Notes: See also enable non-standard constants.

3.1.2.2.10 -ecr* Allow constant ranges

This command-line compiler option enables/disables constant ranges.

Syntax: **-ecr[+|-]**
Default: Enabled
Notes: See also allow constant ranges.

3.1.2.2.11 -edq* Allow double-quoted literals

This command-line compiler option enables/disables double quoted literals.

Syntax: **-edq[+|-]**
Default: Enabled
Notes: See also allow double-quoted (") literals.

3.1.2.2.12 -efn* Enable non-standard functions

This command-line compiler option enables/disables non-standard functions.

Syntax: **-efn[+|-]**
Default: Enabled
Notes: See also enable non-standard functions.

3.1.2.2.13 -enn* Allow non-numeric labels

This command-line compiler option enables/disables support for non-numeric statement labels.

Syntax: **-enn[+|-]**
Default: Enabled
Notes: See also allow non-numeric labels.

3.1.2.2.14 -eop* Enable non-standard operators

This command-line compiler option enables/disables support for non-standard operators.

Syntax: **-eop[+|-]**
Default: Enabled
Notes: See also enable non-standard operators.

3.1.2.2.15 -eow* Allow otherwise

This command-line compiler option enables/disables support for the keyword **otherwise**.

```
Syntax:  -eow[+|-]
Default: Enabled
Notes: See also allow otherwise.
```

3.1.2.2.16 -epr* Enable non-standard procedures

This command-line compiler option enables/disables non-standard procedures.

```
Syntax:  -epr[+|-]
Default: Enabled
Notes: See also enable non-standard procedures.
```

3.1.2.2.17 -erd* Allow relaxed declarations

This command-line compiler option enables/disables relaxed declarations.

```
Syntax:  -erd[+|-]
Default: Enabled
Notes: See also allow relaxed declarations.
```

3.1.2.2.18 -ety* Enable non-standard types

This command-line compiler option enables/disables non-standard types.

```
Syntax:  -ety[+|-]
Default: Enabled
Notes: See also enable non-standard types.
```

3.1.2.2.19 -eui* Allow underscores in identifiers

This command-line compiler option enables/disables underscores (_) in identifiers.

```
Syntax:  -eui[+|-]
Default: Enabled
Notes: See also allow underscores (_) in identifiers.
```

3.1.2.2.20 -eva* Enable non-standard variables

This command-line compiler option enables/disables non-standard variables.

```
Syntax:  -eva[+|-]
Default: Enabled
Notes: See also enable non-standard variables.
```

3.1.2.2.21 -E* Enable all extensions

This command-line compiler option enables/disables all Irie Pascal extensions to Standard Pascal.

```
Syntax:  -E[+|-]
Default: Enabled
```

3.1.2.2.22 -gs Generate a WinNT/2000 service

This command-line compiler option makes the compiler generate a Windows NT/2000 service application.

```
Syntax:  -gs[+|-]
Default: Disabled
Notes: See also generate Windows NT/2000 service application.
```

3.1.2.2.23 -hTEXT Add #!TEXT header

This command-line compiler option specifies a #! header to be inserted in front of the generated executable.

```
Syntax:  -hTEXT(Sets the #! header to TEXT)
Default: By default no #! header is used.
NOTES: See also #! header.
```

3.1.2.2.24 -i* Trap I/O errors

This command-line compiler option enables/disables I/O trapping.

```
Syntax:  -i[+|-]
Default: Enabled
Notes: When I/O trapping is enabled the compiler generates code which checks each I/O
operation and issues a run-time error if an I/O error is detected.
```

3.1.2.2.25 -I Control informatory messages

This command-line compiler option enables/disables information messages.

Syntax: **-I[nn][+|-]**

Default: All information message enabled.

- Use **-I** or **-I+** to enable all information messages.
- Use **-I-** to disable all information messages.
- Use **-Inn** or **-Inn+** to enable the information message with message number **nn**.
- Use **-Inn-** to disable the information message with message number **nn**.

3.1.2.2.26 -ln* Insert line-number debug info

This command-line compiler option enables/disables line number debugging information.

```
Syntax:  -ln[+|-]
Default: Enabled
Notes: See also insert line-number debugging information.
```

3.1.2.2.27 -mb Generate Borland compatible messages

This command-line compiler option enables/disables Borland compatible messages.

Syntax: **-mb[**+|**-]**

Default: Disabled

Notes: When this compiler option is enabled the compiler displays messages in the format used by Borland compilers. This option can be used to make it easier to integrate the Irie Pascal command-line compiler with third-party editors and IDE's which already know how to process Borland compiler messages. This option automatically suppresses the mc compiler option.

3.1.2.2.28 -mc* Display message context

This command-line compiler option enables/disables message context information.

Syntax: **-mc[**+|**-]**

Default: Enabled

Notes: When this compiler option is enabled the compiler shows the position in the source file referred to by error and warning messages. NOTE: Some error and warning messages do not refer to a particular position in the source file and therefore context information is not displayed for those messages.

3.1.2.2.29 -meN Set maximum errors

This command-line compiler option specifies the maximum number of error messages that the compiler should allow.

Syntax: **-meN**

Default: N = 25

Notes: Suppose you want the compiler to stop after 5 error messages then use:

-me5

3.1.2.2.30 -mm Generate Microsoft compatible messages

This command-line compiler option enables/disables Microsoft compatible messages.

Syntax: **-mm[+|-]**

Default: Disabled

Notes: When this compiler option is enabled the compiler displays messages in the format used by Microsoft compilers. This option can be used to make it easier to integrate the Irie Pascal command-line compiler with third-party editors and IDE's which already know how to process Microsoft compiler messages. This option automatically suppresses the mc compiler option.

3.1.2.2.31 -mwN Set maximum warnings

This command-line compiler option specifies the maximum number of warning messages that the compiler should allow.

Syntax: **-mwN**

Default: N = 100

Notes: Suppose you want the compiler to stop after 5 warning messages then use

-mw5

3.1.2.2.32 -nc Allow nested comments

This command-line compiler option enables/disables support for nested comments.

Syntax: **-nc[+|-]**

Default: Disabled

Notes: See also allow nested comments.

3.1.2.2.33 -nu Non-standard unary operators

This command-line compiler option enables/disables non-standard use of unary plus and minus operators.

Syntax: **-nu[+|-]**

Default: Enabled

Notes: See also non-standard unary operators.

3.1.2.2.34 -oNAME Set output filename

This command-line compiler option specifies the name of the executable generated by the compiler.

Syntax: **-oNAME**

Default: The name of the executable is the name of the Pascal source file with the extension changed to **.ivm**.

3.1.2.2.35 -p Require parentheses

This command-line compiler option enables/disables mandatory parentheses mode.

Syntax: **-p[+|-]**

Default: Disabled

Notes: See also require parentheses.

3.1.2.2.36 -r* Trap range errors

This command-line compiler option enables/disables range checking.

Syntax: **-r[+|-]**

Default: Enabled

Notes: See also values out of range.

3.1.2.2.37 -rtlf Send run-time errors to log file

This command-line compiler option sends run-time errors to a log file.

Syntax: **-rtlf[+|-]**

Default: Disabled

Notes: See also run-time errors in log file.

3.1.2.2.38 -rtmb* Send run-time errors to message box

This command-line compiler option sends run-time errors to a message box. NOTE: Only executables running under the Windows platform can display run-time errors in message boxes, on all other platforms this option has no effect.

Syntax: **-rtmb[+|-]**

Default: Enabled

Notes: See also run-time errors in message box.

3.1.2.2.39 -rtsc* Send run-time errors to screen

This command-line compiler option sends run-time errors to the console screen.

Syntax: **-rtsc[+|-]**

Default: Enabled

Notes: See also run-time errors to console screen.

3.1.2.2.40 -s* Strict var strings

This command-line compiler option enables/disables strict checking of var string parameters.

Syntax: **-s[+|-]**

Default: Enabled

Notes:

When strict checking is enabled, it is an error to pass a string variable by reference if the length of the variable's string type is not equal to the length of the formal parameter's string type.

When strict checking is disabled you can pass a string variable by reference even if the length of the string variable is not equal to the length of the formal parameter.

For example if you compile the following program and strict checking is enabled:

```
program p(output);
type
string16 = string[16];
string8 = string[8];
var
x : string16;
procedure print(var s : string8);
begin
writeln(s)
end;
begin
x := 'Hello';
print(x)          (* Error only if strict checking is enabled *)
end.
```

then the compiler will report an error with the call **print(x)** since the length of **x's** string type is 16 and the length of the formal parameter's sting type is 8 (i.e. they are not equal).

If strict checking is disabled then no errors are reported.

3.1.2.2.41 -sc* Use short-circuit evaluation

This command-line compiler option enables/disables short-circuit evaluation for the boolean operators **and** and **or**.

Syntax: **-sc[+|-]**

Default: Enabled

Notes: See also use short-circuit evaluation.

3.1.2.2.42 -so Maximum stack overflow checking

This command-line compiler option enables/disables detailed stack overflow checking.

Syntax: **-so[+|-]**

Default: Disabled

Notes: The interpreter automatically checks for stack overflow at the beginning and end of each procedure/function call and when large values are placed on the stack. These checks should suffice for most purposes, however you can enable this option if you want the interpreter to check for stack overflow before every statement is executed.

3.1.2.2.43 -Snn Set stack size in K

This command-line compiler option specifies the number of kilobytes to allocate for your program's stack.

Syntax: **-Snn**

Default: nn = 64

Notes: See also specify stack size.

3.1.2.2.44 -u* Trap use of undefined values

This command-line compiler option enables/disables checking for undefined values.

Syntax: **-u[+|-]**

Default: Enabled

Notes: When this option is enabled the compiler generates code which checks each time your program gets a value from a variable to make sure that the value is not undefined. If your program does get an undefined value from a variable then the code generated by the compiler will issue a run-time error message and terminate your program.

So for example if you compile and run the following program

```
program bad(output);
var
r : real;
begin
writeln(r) (* The value of "r" is undefined *)
end.
```

The program will terminate with a run-time error message because of the attempt to print the value of **r** which is undefined. Unfortunately not all variable accesses can be checked, checks are only made for accesses to variables of the following types:

- enumerated types (including boolean)
- subranges of enumerated types
- subranges of integer that do not include -1
- file
- list
- object
- pointer
- real
- set (array representation)

Accesses to variables of types **char**, **integer**, **record** and **set** (Bit set representation) are not checked.

Checking for undefined values is performed as follows:

- All memory is initialized by setting all bits to 1.
- When accessing a variable a check is performed to see if all bits are set and if they are this variable is undefined. This doesn't work for **char**, **integer** or **set** (bit set representation) variables since a value with all bits set to 1 is valid for these variables. Record variables are not checked because they may contain fields which can not be checked.

3.1.2.2.45 -v* Trap use of inactive variants

This command-line compiler option enables/disables variant checking.

Syntax: **-v[+|-]**

Default: Enabled

Notes: When variant checking is enabled the compiler generates code to check each time a field of a variant, is accessed, to make sure that the variant is active.

3.1.2.2.46 -W Control warning messages

This command-line compiler option enables/disables warning messages.

Syntax: **-W[nn][+|-]**

Default: All warning messages enabled.

- Use **-W** or **-W+** to enable all warning messages.
- Use **-W-** to disable all warning messages.
- Use **-Wnn** or **-Wnn+** to enable warning message number **nn**.

- Use **-Wnn-** to disable warning message number **nn**.

3.2.1 Using the interpreter

The Irie Virtual Machine Interpreter is used to run Irie Virtual Machine executables. The Linux, FreeBSD, Solaris/x86 and Solaris/Sparc editions of Irie Pascal contain two different versions of the interpreter (**ivm** and **ivmdb**). The only difference between these two versions is that **ivm** does not include support for database programming and will run whether or not you have MySQL installed, while **ivmdb** does include support for database programming (currently only MySQL databases are supported) but requires MySQL to be installed before it will run.

The interpreter should be run from a command-line prompt.

Once you are at a command-line prompt use the following syntax:

Syntax: **ivm [filename] [arguments]**

where **filename** specifies the Irie Virtual Machine Executable to run.

and **[arguments]** are program arguments passed to the executable.

NOTE: **[x]** indicates that **x** is optional.

For example if you compile the sample program **hello.pas**, the compiler will generate a file called **hello.ivm** which contains an Irie Virtual Machine Executable.

You can run it by entering **ivm ./hello** or **ivm ./hello.ivm**.

3.3.1 Using the header utility

NOTE: The command-line and IDE compilers can put #! headers into the generated executables so using the Irie Header Utility is not necessary anymore unless you can't or don't want to recompile the executable.

The Irie Header Utility is used to create IVM executables with #! headers. #! headers can be used by UNIX or UNIX-like operating systems (such as Solaris, Linux or FreeBSD) to locate the interpreters that should be used to execute scripts. When UNIX or a UNIX-like operating system attempts to execute a file and the first two characters in the file are #!, the operating system will assume that the file is a script that needs to be interpreted, and it will expect the location of the interpreter to follow the #!.

The syntax is:

```
ivm header input-executable output-executable [location]
```

where

- **input-executable** - is the ivm executable you want to use to create the new executable from.
- **output-executable** - is the ivm executable you want to create
- **location** - if specified is the location of the interpreter to put in the #! header.

So for example if the interpreter was installed in **/usr/local/bin**, you would store **#!/usr/local/bin/ivm** in front of the executable.

So for example if you had an IVM executable named **hello.ivm** you could enter

```
ivm header hello.ivm hello /usr/local/bin/ivm
```

which creates a new executable called **hello** which has

```
#!/usr/local/bin/ivm
```

in front (#! is automatically inserted). In order to be able to execute **hello** you also need to set the executable permission bit (using

```
chmod a+x hello
```

for example).

4.1 Overview of extensions to Standard Pascal

Standard Pascal (i.e. ISO/IEC 7185) allows implementations, such as Irie Pascal, to add new features, called extensions. However an extension to Standard Pascal can not invalidate any Standard Pascal program that would otherwise be valid, except possibly by adding new keyword(s) that would invalidate programs that used those keyword(s) as normal identifiers. **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

Irie Pascal supports a number of extensions to Standard Pascal. Some of these extensions were added for compatibility with Turbo Pascal or Extended Pascal, while others were added for other reasons and are likely to be specific to Irie Pascal.

The Irie Pascal extensions to Standard Pascal are listed below:

- Auto declare input and output
- Allow binary/hexadecimal constants
- Enable non-standard constants
- Allow constant ranges
- Allow double-quoted literals
- Enable non-standard functions
- Allow non-numeric labels
- Enable non-standard operators
- Allow 'otherwise'
- Enable non-standard procedures
- Allow relaxed declaratons
- Enable non-standard types
- Allow underscores (_) in identifiers
- Enable non-standard variables

"4.2 Auto declare input and output"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will automatically declare the built-in identifiers **input** and **output**, when they are not declared in your program.

Standard Pascal (i.e. ISO/IEC 7185) specifies that whenever the required identifiers **input** and **output** are referenced in a program, that they must by declared (i.e. appear as program parameters). However because some Pascal compilers do not enforce this requirement many Pascal programs do not meet this specification. **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

"4.3 Allow binary/hexadecimal constants"

When this extension is enabled (the **default**), the Irie Pascal compiler will recognize binary and hexadecimal constants.

Binary Constants

Binary constants begin with **%**, and are followed by one or more binary (**0** or **1**) digits.

The following are examples of valid binary constants

```
%0    %1       %01110101010101010111101
```

The following are not valid binary constants

```
%2     %       %151        %g
```

Hexadecimal Constants

Hexadecimal constants begin with **$**, and are followed by one or more hexadecimal (**one of 0123456789ABCDEF**) digits.

The following are examples of valid hexadecimal constants

```
$9    $A123       $ffff
```

The following is not a valid hexadecimal constant

```
$abgd
```

since **g** is not a hexadecimal digit.

"4.4 Enable non-standard constants"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will recognized a number of non-standard constants, which are not part of Standard Pascal (i.e. ISO/IEC 7185). **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for a complete list of non-standard constants.

"4.5 Allow constant ranges"

When this extension is enabled (the **default**), the Irie Pascal compiler will recognize constant ranges in **case statements** and **variant records**.

Constant Ranges

You can use constant ranges to specify a number of consecutive case constants. To use a constant range you specify the first constant, and the last constant, separated by **..** as follows:

```
first..last
```

So for example you could use the constant range

```
1..5
```

to specify the following constants

```
1, 2, 3, 4, 5
```

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information.

"4.6 Allow double-quoted literals"

When this extension is enabled (the **default**), the Irie Pascal compiler will allow you to use double-quotes to deliminate character and string literals.

For example when this extension is enabled you could use

```
"Hello world"
```

instead of

```
'Hello world'
```

Double-quoted literals can be especially useful if you want to create literals with single quotes in them, since you don't have to use two single quotes to represent one single quote.

For example you could use

```
"Don't go away"
```

which is equivalent to

```
'Don''t do away'
```

but much more readable.

"4.7 Enable non-standard functions"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will recognized a number of non-standard functions, which are not part of Standard Pascal (i.e. ISO/IEC 7185). **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for a complete list of non-standard functions.

"4.8 Allow non-numeric labels"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will allow non-numeric statement labels. In Standard Pascal (i.e. ISO/IEC 7185), statement labels must be numeric and between 0 and 9999. When this extension is enabled, you can declare and use labels containing letters and underscores. **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by

the Internation Organization for Standardization.

For example in the following program, **loop** is used as a statement label.

```
program name(output);
label loop;
var
i : integer;
begin
i := 1;
loop:
writeln(i);
i := i + 1;
if i <= 20 then goto loop;
end.
```

"4.9 Enable non-standard operators"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will recognize a number of non-standard operators, which are not a part of Standard Pascal (i.e. ISO/IEC 7185), and it will allow some of the boolean operators to be used to perform bitwise operations. **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

The non-standard operators are:

```
and_then, or_else, xor, shl, shr
```

The boolean operators that can perform bitwise operations are:

```
not, and, or
```

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information.

"4.10 Allow 'otherwise'"

When this extension is enabled (the **default**), the Irie Pascal compiler will allow you to use the keyword **otherwise** in case statements and variant records to specify "all values that haven't been used yet". You can also use the keyword **else** instead of **otherwise** (this feature was added in order to improve compatibility with Turbo Pascal).

"4.11 Enable non-standard procedures"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will recognized a number of non-standard procedures, which are not part of Standard Pascal (i.e. ISO/IEC 7185). **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for a complete list of non-standard procedures.

"4.12 Allow relaxed declaratons"

Standard Pascal (i.e. ISO/IEC 7185) requires that all declarations/definitions of the same kind must be made together in a single group and that the groups must appear in a specific order. **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for

Standardization.

The order of declarations/definitions required by Standard Pascal is:

```
Label declaration group
Constant definition group
Type definition group
Variable declaration group
sub-block declaration group
```

When this extension is enabled (it is by **default**), there can be more than one of each kind of group and groups can appear in any order except that, for declarations local to a function or procedure, the sub-block declaration group must be last.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more information about declarations/definitions.

"4.13 Enable non-standard types"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will recognized a number of non-standard types, which are not part of Standard Pascal (i.e. ISO/IEC 7185). **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for more a complete list of non-standard types.

"4.14 Allow underscores (_) in identifiers"

When this extension is enabled (the **default**), the Irie Pascal compiler will allow underscores in identifiers.

So for example the following would be valid identifiers:

```
_name
last_name
_all_names_
```

"4.15 Enable non-standard variables"

When this extension is enabled (it is by **default**), the Irie Pascal compiler will recognized a number of non-standard variables, which are not part of Standard Pascal (i.e. ISO/IEC 7185). **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

See the Irie Pascal Programmer's Reference Manual (in "progref.html") for a complete list of non-standard variables.

5.1 What is Irie Pascal?

Irie Pascal is a Pascal compiler and interpreter. The compiler translates Pascal programs into Irie Virtual Machine (IVM) executables, which are then executed by the interpreter. The IVM is an abstract computer platform that is implemented in software (by the interpreter), and runs executables on many different

computer platforms. The IVM has been implemented on the follwing computer platforms (Win95/98/NT/2000/XP, Linux, FreeBSD, Solaris/x86, and Solaris/Sparc) so far. IVM executables developed on any platform, run on all the other platforms.

Irie Pascal's ability to generate executables which run on multiple platforms make it ideally suited for creating internet applications. The Common Gateway Interface (CGI) is a simple but powerful protocol for creating server side internet applications. Irie Pascal assists the creation of CGI scripts with built-in support for decoding and parsing URL encoded strings, as well as support for databases, and sending email. Irie Pascal also supports the UNIX #! trick that allows the location of the interpreter to be embedded inside the script making it easier to execute the script from a URL, since the URL need only refer to the script and not the interpreter.

Irie Pascal is highly compatible with Standard Pascal (i.e. ISO/IEC 7185). This high level of compatibility means that Irie Pascal shares Standard Pascal's strengths as a first language for beginners. These strengths include readable syntax, and extensive compile-time and run-time checking.
NOTE: ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

Irie Pascal supports many extensions to Standard Pascal, particularly in the areas of string, file/folder processing, and database programming, which make it useful for creating scripts and utilities. Irie Pascal's support for automatic run-time checking make it useful for creating **quick and dirty** programs (i.e. programs that are expected to be run only a few times or by only a few people and may not be worth spending a lot of time on).

5.2 Compliance

Irie Pascal complies with the requirements of level 0 of ISO/IEC 7185, with the following exceptions: (see the Irie Pascal Reference Manual, Appendix B - Deviations from ISO/IEC 7185). **NOTE:** ISO/IEC 7185 is the standard for the Pascal porgramming language published by the Internation Organization for Standardization.

NOTE: Irie Pascal compliance with ISO/IEC 7185 has not been formally certified by an external body.

5.3 License and distribution rights

IRIE PASCAL EVALUATION VERSION (SOLARIS/SPARC EDITION)

LICENSE STATEMENT AND DISCLAIMER OF WARRANTY

IMPORTANT - READ CAREFULLY This license statement and disclaimer of warranty constitutes a legal agreement ("License Agreement") between you (either as an individual or a single entity) and Stuart King ("Author") for the software product ("Software") identified above, including any software, media, and accompanying on-line or printed documentation.

BY INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE, YOU AGREE TO BE BOUND BY ALL OF THE TERMS AND CONDITIONS OF THE LICENSE AGREEMENT.

Upon your acceptance of the terms and conditions of the License Agreement, the Author grants you the right to use the Software in the manner provided below.

This Software is owned by the Author and is protected by copyright law and international copyright treaty. Therefore, you must treat this Software like any other copyrighted material (e.g. a book), except

that you may either make one copy of the Software solely for backup or archival purposes or transfer the Software to a single hard disk provided you keep the original solely for backup or archival purposes.

The Author grants to you as an individual, a personal, nonexclusive, non-transferable license to install and use the Software for evaluation purposes only. In particular, you may not distribute or cause to be distributed the Software or any programs you develop using the Software. You may install a copy of the Software on a computer and freely move the Software from one computer to another, provided that you are the only individual using the Software. If you are an entity, the Author grants you the right to designate one individual within your organization ("Named User") to have the right to use the Software in the manner provided above.

The Software might include source code, redistributable files, and/or other files provided by a third party vendor (Third Party Software). Since use of Third Party Software might be subject to license restrictions imposed by the third party vendor, you should refer to the on-line documentation (if any) provided with Third Party Software for any license restrictions imposed by the third party vendor. In any event, any license restrictions imposed by a third party vendor are in addition to, not in lieu of, the terms and conditions of the License Agreement.

DISCLAIMER OF WARRANTY

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THE AUTHOR DISCLAIMS ALL WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. THIS WARRANTY DISCLAIMER AFFECTS YOUR LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM JURISDICTION TO JURISDICTION. SOME JURISDICTIONS DO NOT ALLOW EXCLUSIONS OF IMPLIED WARRANTIES OR LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU.

HIGH RISK ACTIVITIES The Software is not fault-tolerant and is not designed, manufactured or intended for use or resale as on-line control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of the Software could lead directly to death, personal injury, or severe physical or environmental damage ("High Risk Activities"). The Author specifically disclaims any express or implied warranty of fitness for High Risk Activities.

LIMITATION OF LIABILITY

IN NO EVENT SHALL THE AUTHOR BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY CONSEQUENTIAL, SPECIAL, INCIDENTAL OR INDIRECT DAMAGES OF ANY KIND ARISING OUT OF OR RELATING TO THE SOFTWARE OR THE USE THEREOF (INCLUDING BUT NOT LIMITED TO LOST PROFITS OR OTHER ECONOMIC LOSS), EVEN IF THE AUTHOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT WILL THE AUTHOR'S LIABILITY FOR ANY CLAIM, WHETHER IN CONTRACT, TORT, OR ANY OTHER THEORY OF LIABILITY, EXCEED THE FEE PAID BY YOU FOR THE SOFTWARE THAT IS THE SUBJECT OF SUCH CLAIM. IF THE RELEVANT SOFTWARE WAS PROVIDED TO YOU AT NO CHARGE YOU AGREE THAT THE AUTHOR SHALL NOT BE LIABLE TO YOU FOR ANY DAMAGES. YOU AGREE THAT YOU ARE SOLELY RESPONSIBLE FOR ADEQUATE PROTECTION AND BACKUP OF THE DATA AND EQUIPMENT USED IN CONNECTION WITH THE SOFTWARE OR SUBSCRIPTION SERVICES, AND FURTHER AGREE THAT THE AUTHOR WILL NOT BE LIABLE FOR ANY DAMAGES THAT YOU MAY SUFFER IN CONNECTION WITH DOWNLOADING, INSTALLING, OR USING THE SOFTWARE. IF YOU ELECT NOT TO PURCHASE A LICENSE TO THE SOFTWARE, YOU FURTHER ACKNOWLEDGE THAT YOU

ARE PROVIDED A REASONABLE TIME FRAME TO EVALUATE THE SOFTWARE AND AT THE END OF SUCH EVALUATION PERIOD YOU MAY ONLY ACCESS AND USE THE SOFTWARE IF YOU PURCHASE A LICENSE TO THE SOFTWARE. YOU AGREE THAT THE AUTHOR WILL NOT BE LIABLE FOR ANY DAMAGE THAT YOU MAY SUFFER IN CONNECTION WITH THE TERMINATION OF SUCH EVALUATION PERIOD AND YOUR INABILITY TO ACCESS AND USE THE SOFTWARE. THIS LIMITATION SHALL APPLY TO CLAIMS OF PERSONAL INJURY TO THE EXTENT PERMITTED BY LAW. THE LIMITATIONS IN THIS SECTION ARE SEPARATE AND INDEPENDENT OF ANY OTHER REMEDY LIMITATIONS IN THIS AGREEMENT AND SHALL NOT FAIL IF SUCH OTHER LIMITATION OR REMEDY FAILS. THE FEES AND OTHER PROVISIONS IN THIS AGREEMENT REFLECT THE ALLOCATION OF RISKS BETWEEN THE PARTIES. THIS SECTION IS AN ESSENTIAL ELEMENT OF THE BASIS OF THE BARGAIN BETWEEN THE PARTIES. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE EXCLUSIONS OR LIMITATIONS MAY NOT APPLY TO YOU.

TERMINATION

This Agreement shall terminate automatically if you fail to comply with the terms of this Agreement. This Agreement shall terminate if you do not purchase a license to the Software within a period of 30 calender days from the date the Software is first installed by you. No notice shall be required from the Author to effect such termination. You may also terminate this Agreement at any time by uninstalling and destroying all copies of the Software.

ENTIRE AGREEMENT

You agree that this is the entire agreement between you and the Author, and that it supersedes any prior agreement, whether written or oral, and all other communications between the Author and you relating to the subject matter of this Agreement. This Agreement may be amended, modified or supplemented only by a writing that is signed by the authorized representatives of both parties.

RESERVATION OF RIGHTS

5.4 Disclaimer-Agreement

See the license statement for the warranty disclaimer.

5.5.1 Getting help from the manuals

The Irie Pascal User's Manual

The Irie Pascal User's Manual (in "user.html"), which is the manual you are currently reading, provides help on using Irie Pascal.

The Programmer's Reference Manual

The Irie Pascal Programmer's Reference Manual (in "progref.html") provides help on the Irie Pascal programming language.

5.5.2 Getting help from the website

The Irie Tools Website

The Irie Tools website is an online source of up-to-date information about Irie Pascal. You can access this website at www.irietools.com.

5.5.3 Contacting customer support

Email Help

Irie Pascal help is also available at support@irietools.com. Queries to this email address are normally answered within 24 hours.

5.6.1 Checking prices

Irie Pascal is available in a number of different editions, and at the time this help file was created the available editions were: **Windows**, **Linux**, **FreeBSD**, **Solaris/x86**, **Solaris/Sparc**, and **Universal**. The price for Irie Pascal licenses varies depending on the edition, the number of licenses, and the currency.

The Irie Tools website is an online source of up-to-date pricing information. You can access this website at www.irietools.com. Or you can view the pricing information contained in this Manual.

5.6.2.1 Irie Pascal Windows Edition Prices (in US$)

No. Users	Unit Price (US$)	Total Price (US$)
1	79.99	79.99
5	69.99	349.95
10	59.99	599.90

No. Users	Unit Price (US$)	Total Price (US$)
20	49.99	999.80
40	39.99	1,599.60
100	29.99	2,999.00

All prices are quoted in US$ and are guaranteed until December 31, 2005.

5.6.2.2 Irie Pascal Windows Edition Prices (in CA$)

No. Users	Unit Price (CA$)	Total Price (CA$)
1	99.99	99.99
5	87.49	437.45
10	74.99	749.90
20	62.49	1,249.80
40	49.99	1,999.60
100	37.49	3,749.00

All prices are quoted in CA$ and are guaranteed until December 31, 2005.

5.6.2.3 Irie Pascal Windows Edition Prices (in UK)

No. Users	Unit Price (UK£)	Total Price (UK£)
1	39.99	39.99
5	34.99	174.95
10	29.99	299.90

No. Users	Unit Price (UK£)	Total Price (UK£)
20	24.99	499.80
40	19.99	799.60
100	14.99	1,499.00

All prices are quoted in UK£ and are guaranteed until December 31, 2005.

5.6.2.4 Irie Pascal Windows Edition Prices (in Euros)

No. Users	Unit Price (Euros)	Total Price (Euros)
1	59.99	59.99
5	52.99	264.95
10	45.99	459.90
20	38.99	779.80
40	31.99	1,279.60
100	24.99	2,499.00

All prices are quoted in Euros and are guaranteed until December 31, 2005.

5.6.3.1 Irie Pascal Linux Edition Prices (in US$)

No. Users	Unit Price (US$)	Total Price (US$)
1	79.99	79.99
5	69.99	349.95

No. Users	Unit Price (US$)	Total Price (US$)
10	59.99	599.90
20	49.99	999.80
40	39.99	1,599.60
100	29.99	2,999.00

All prices are quoted in US$ and are guaranteed until December 31, 2005.

5.6.3.2 Irie Pascal Linux Edition Prices (in CA$)

No. Users	Unit Price (CA$)	Total Price (CA$)
1	99.99	99.99
5	87.49	437.45
10	74.99	749.90
20	62.49	1,249.80
40	49.99	1,999.60
100	37.49	3,749.00

All prices are quoted in CA$ and are guaranteed until December 31, 2005.

5.6.3.3 Irie Pascal Linux Edition Prices (in UK)

No. Users	Price (UK£)	Total Price (UK£)
1	39.99	39.99

No. Users	Price (UK£)	Total Price (UK£)
5	34.99	174.95
10	29.99	299.90
20	24.99	499.80
40	19.99	799.60
100	14.99	1,499.00

All prices are quoted in UK£ and are guaranteed until December 31, 2005.

5.6.3.4 Irie Pascal Linux Edition Prices (in Euros)

No. Users	Unit Price (Euros)	Total Price (Euros)
1	59.99	59.99
5	52.99	264.95
10	45.99	459.90
20	38.99	779.80
40	31.99	1,279.60
100	24.99	2,499.00

All prices are quoted in Euros and are guaranteed until December 31, 2005.

5.6.4.1 Irie Pascal FreeBSD Edition Prices (in US$)

No. Users	Unit Price (US$)	Total Price (US$)
1	79.99	79.99
5	69.99	349.95
10	59.99	599.90
20	49.99	999.80
40	39.99	1,599.60
100	29.99	2,999.00

All prices are quoted in US$ and are guaranteed until December 31, 2005.

5.6.4.2 Irie Pascal FreeBSD Edition Prices (in CA$)

No. Users	Unit Price (CA$)	Total Price (CA$)
1	99.99	99.99
5	87.49	437.45
10	74.99	749.90
20	62.49	1,249.80
40	49.99	1,999.60
100	37.49	3,749.00

All prices are quoted in CA$ and are guaranteed until December 31, 2005.

5.6.4.3 Irie Pascal FreeBSD Edition Prices (in UK)

No. Users	Price (UK£)	Total Price (UK£)
1	39.99	39.99
5	34.99	174.95
10	29.99	299.90
20	24.99	499.80
40	19.99	799.60
100	14.99	1,499.00

All prices are quoted in UK£ and are guaranteed until December 31, 2005.

5.6.4.4 Irie Pascal FreeBSD Edition Prices (in Euros)

No. Users	Unit Price (Euros)	Total Price (Euros)
1	59.99	59.99
5	52.99	264.95
10	45.99	459.90
20	38.99	779.80
40	31.99	1,279.60
100	24.99	2,499.00

All prices are quoted in Euros and are guaranteed until December 31, 2005.

5.6.5.1 Irie Pascal Solaris/x86 Edition Prices (in US$)

No. Users	Unit Price (US$)	Total Price (US$)
1	79.99	79.99
5	69.99	349.95
10	59.99	599.90
20	49.99	999.80
40	39.99	1,599.60
100	29.99	2,999.00

All prices are quoted in US$ and are guaranteed until December 31, 2005.

5.6.5.2 Irie Pascal Solaris/x86 Edition Prices (in CA$)

No. Users	Unit Price (CA$)	Total Price (CA$)
1	99.99	99.99
5	87.49	437.45
10	74.99	749.90
20	62.49	1,249.80
40	49.99	1,999.60
100	37.49	3,749.00

All prices are quoted in CA$ and are guaranteed until December 31, 2005.

5.6.5.3 Irie Pascal Solaris/x86 Edition Prices (in UK)

No. Users	Price (UK£)	Total Price (UK£)
1	39.99	39.99
5	34.99	174.95
10	29.99	299.90
20	24.99	499.80
40	19.99	799.60
100	14.99	1,499.00

All prices are quoted in UK£ and are guaranteed until December 31, 2005.

5.6.5.4 Irie Pascal Solaris/x86 Edition Prices (in Euros)

No. Users	Unit Price (Euros)	Total Price (Euros)
1	59.99	59.99
5	52.99	264.95
10	45.99	459.90
20	38.99	779.80
40	31.99	1,279.60
100	24.99	2,499.00

All prices are quoted in Euros and are guaranteed until December 31, 2005.

5.6.6.1 Irie Pascal Solaris/Sparc Edition Prices (in US$)

No. Users	Unit Price (US$)	Total Price (US$)
1	79.99	79.99
5	69.99	349.95
10	59.99	599.90
20	49.99	999.80
40	39.99	1,599.60
100	29.99	2,999.00

All prices are quoted in US$ and are guaranteed until December 31, 2005.

5.6.6.2 Irie Pascal Solaris/Sparc Edition Prices (in CA$)

No. Users	Unit Price (CA$)	Total Price (CA$)
1	99.99	99.99
5	87.49	437.45
10	74.99	749.90
20	62.49	1,249.80
40	49.99	1,999.60
100	37.49	3,749.00

All prices are quoted in CA$ and are guaranteed until December 31, 2005.

5.6.6.3 Irie Pascal Solaris/Sparc Edition Prices (in UK)

No. Users	Price (UK£)	Total Price (UK£)
1	39.99	39.99
5	34.99	174.95
10	29.99	299.90
20	24.99	499.80
40	19.99	799.60
100	14.99	1,499.00

All prices are quoted in UK£ and are guaranteed until December 31, 2005.

5.6.6.4 Irie Pascal Solaris/Sparc Edition Prices (in Euros)

No. Users	Unit Price (Euros)	Total Price (Euros)
1	59.99	59.99
5	52.99	264.95
10	45.99	459.90
20	38.99	779.80
40	31.99	1,279.60
100	24.99	2,499.00

All prices are quoted in Euros and are guaranteed until December 31, 2005.

5.6.7.1 Irie Pascal Universal Edition Prices (in US$)

No. Users	Unit Price (US$)	Total Price (US$)
1	149.99	149.99
5	131.99	659.95
10	113.99	1,139.90
20	95.99	1,919.80
40	77.99	3,119.60
100	59.99	5,999.00

All prices are quoted in US$ and are guaranteed until December 31, 2005.

5.6.7.2 Irie Pascal Universal Edition Prices (in CA$)

No. Users	Unit Price (CA$)	Total Price (CA$)
1	187.49	187.49
5	164.99	824.95
10	142.49	1,424.90
20	119.99	2,399.80
40	97.49	3,899.60
100	74.99	7,499.00

All prices are quoted in CA$ and are guaranteed until December 31, 2005.

5.6.7.3 Irie Pascal Universal Edition Prices (in UK)

No. Users	Unit Price (UK£)	Total Price (UK£)
1	74.99	74.99
5	65.99	329.95
10	56.99	569.90
20	47.99	959.80
40	38.99	1,559.60
100	29.99	2,999.00

All prices are quoted in UK£ and are guaranteed until December 31, 2005.

5.6.7.4 Irie Pascal Universal Edition Prices (in Euros)

No. Users	Unit Price (Euros)	Total Price (Euros)
1	112.49	112.49
5	98.99	494.95
10	85.49	854.90
20	71.99	1,439.80
40	58.49	2,339.60
100	44.99	4,499.00

All prices are quoted in Euros and are guaranteed until December 31, 2005.

5.7.1 Why you should buy a license

There are basically two reasons why you should buy an Irie Pascal license.

- The first reason is that it is the right thing to do. When you buy an Irie Pascal license you are purchasing the legal right to use Irie Pascal under the terms specified in the license.
- The second reason is that it is in your own enlightened self-interest. When you buy an Irie Pascal license you contribute to the continued development of Irie Pascal. If you use Irie Pascal you are making an investment of your time and effort, and as Irie Pascal develops the value of your investment increases (i.e. you can do more of what you want to do with less effort).

So if you find Irie Pascal useful, it makes sense to buy a license to use it.

5.7.2 How do you buy a license

Buying an Irie Pascal license is quick, easy, and secure. You can choose from a number of options when making your purchase. These options are described below.

- **Payment Methods:**

  You can buy Irie Pascal licenses using a variety of payment methods (Credit Card, Check/Money Order, Wire Transfer, Purchase Order).

- **Payment Sites:**

  Payments for Irie Pascal licenses are accepted at a number of different sites. You can make your purchase directly to Irie Tools by fax or mail. You can also purchase from one of the following payment processors, ShareIt (at www.ShareIt.com), or RegNow (at www.RegNow.com).

- **Currencies:**

  You can choose to pay for licenses in a variety of currencies (US$, CA$, UK£, and Euros).

Buying Licenses:

Not all combinations of options are possible (for example all purchase orders must be sent directly to Irie Tools and not to the other payment processors). For this reason it is recommended that you allow the Irie Tools website to guide you through the buying process. Don't worry it is easier to buy a license than it is to describe how to buy a license.

From the Irie Tools website (at www.irietools.com/iriepascal/buy.html), you will be able to have an order form generated and emailed to you (you would then complete the order form and fax it in or mail it in along with your payment). From the website you will also be able to go to one of the online credit card processors which can accept your payment. The online credit card processors have been carefully selected to ensure that your payment information will be secure, and allow you to download Irie Pascal immediately after your payment is processed.

Sales enquiries can be sent to sales@irietools.com.

5.7.3 Buying through the web

The Irie Tools website at www.irietools.com/iriepascal/buy.html can guide you through the process of buying an Irie Pascal license.

From the website you can have an order form generated (which you can print using your browser). You can have the order form emailed to you (you can then print it using your email program). After printing the order form you would then complete it and fax it (see buying by fax for more information) or mail it (see buying by mail) for more information.

From the website you can be transferred to one of the online payment processors which can accept your payment. The online payment processors have been carefully selected to ensure that your payment information will be secure. If you pay through an online payment processor then you will have the option of downloading Irie Pascal immediately after your payment is processed.

Sales enquiries can be sent to sales@irietools.com.

5.7.4 Buying by telephone

If you are paying by credit card then it is possible to phone in your order. Telephone orders are accepted by ShareIt (one of the online payment processors authorized to accept payments for Irie Pascal licenses). You can access ShareIt through the Irie Tools website at www.irietools.com/iriepascal/buy.html or you can go directly to ShareIt at www.ShareIt.com and search for Irie Pascal.

NOTE: ShareIt has been carefully selected as an authorized payment processor to ensure that your payment information will be secure.

Sales enquiries can be sent to sales@irietools.com.

5.7.5 Buying by Fax

If you are paying with a credit card or if you are sending in a purchase order then it is possible to fax in your order.

First you will need to get an order form. You can get an order form from the Irie Tools website (at www.irietools.com/iriepascal/buy.html), or by printing an appripriate page from this manual. If you get an order from the Irie Tools website, you will be able to have an order form generated for you, which you can then print or have it emailed to you.

If you are paying with a credit card then you will need to fill in the order form and fax it to ShareIt (one of the online payment processors authorized to accept payments for Irie Pascal licenses). Go to www.ShareIt.com.

If you sending in a purchase order then you must fax it along with the completed order form. **NOTE:** If you are not already a customer of Irie Tools it is best to check whether purchase orders from your company/institution will be accepted before ordering. See Purchase orders for information on using purchase orders.

Sales enquiries can be sent to sales@irietools.com.

5.7.6 Buying through the mail

Irie Pascal orders can be sent in by postal mail.

First you will need to get an order form. You can get an order form from the Irie Tools website (at www.irietools.com/iriepascal/buy.html), or by printing an appripriate page from this manual. If you get an order from the Irie Tools website, you will be able to have an order form generated for you, which you can then print or have it emailed to you.

Then you will need to fill in the order form and mail it to:

```
Attn: Stuart King
Irie Tools
221 S. State Road 7, MB #247
Plantation, FL, 33317, USA
```

If you are using a purchase order then you must mail it along with the completed order form. NOTE: If you are not already a customer of Irie Tools it is best to check whether purchaseorders from your company/institution will be accepted before ordering. See Purchase orders for information on using purchase orders.

If you are using a check or money order then you should make it out to **Irie Tools** and mail it along with the completed order form.

Sales enquiries can be sent to sales@irietools.com.

5.7.7 Buying by wire transfer

If you are paying by wire transfer card then you should order through ShareIt (one of the online payment processors authorized to accept payments for Irie Pascal licenses). You can access ShareIt through the Irie Tools website at www.irietools.com or you can go directly to ShareIt at www.ShareIt.com and search for Irie Pascal.

NOTE: ShareIt has been carefully selected as an authorized payment processor to ensure that your payment information will be secure.

Sales enquiries can be sent to sales@irietools.com.

5.7.8 Purchase orders

Purchase orders are accepted from government and accredited educational institutions and major corporations, provided that they are submitted on purchase order forms with a purchase order number. You must include an Irie Pascal order form along with your purchase order. NOTE: If you are not already a customer of Irie Tools it is best to check whether purchase orders from your company/institution will be accepted before ordering.

Purchase orders can be mailed to:

```
Attn: Stuart King
Irie Tools
221 S. State Road 7, MB #247
Plantation, FL, 33317, USA
```

or faxed to **1-876-946-2703**.

Sales enquiries can be sent to sales@irietools.com.

5.7.9.1.1 Irie Pascal (Windows edition) US$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	79.99	_____	$_____
5	69.99	_____	$_____
10	59.99	_____	$_____
20	49.99	_____	$_____
40	39.99	_____	$_____
100	29.99	_____	$_____
Shipping & Handling	5.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed [ ] Charge credit card

[ ] AmEx [ ] MasterCard [ ] Visa

Account #:_____

Expiration Date:_____

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in US$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.1.2 Irie Pascal (Windows Edition) CA$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	99.99	_____	$_____
5	87.49	_____	$_____
10	74.99	_____	$_____
20	62.49	_____	$_____
40	49.99	_____	$_____
100	37.49	_____	$_____
Shipping & Handling	7.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in CA$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.1.3 Irie Pascal (Windows Edition) UK Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	39.99	_____	_____
5	34.99	_____	_____
10	29.99	_____	_____
20	24.99	_____	_____
40	19.99	_____	_____
100	14.99	_____	_____
Shipping & Handling	3.00		_____
TOTAL			_____

No. Users	Unit Price	Quantity	Total Price

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in UK£ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.1.4 Irie Pascal (Windows Edition) Euro Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	59.99	_____	_____
5	52.99	_____	_____
10	45.99	_____	_____
20	38.99	_____	_____
40	31.99	_____	_____
100	24.99	_____	_____

No. Users	Unit Price	Quantity	Total Price
Shipping & Handling	4.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in Euros and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.2.1 Irie Pascal (Linux Edition) US$ Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	79.99	_____	$_____
5	69.99	_____	$_____
10	59.99	_____	$_____
20	49.99	_____	$_____

No. Users	Unit Price	Quantity	Total Price
40	39.99	_____	$_____
100	29.99	_____	$_____
Shipping & Handling	5.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed [ ] Charge credit card

[ ] AmEx [ ] MasterCard [ ] Visa

Account #:_____

Expiration Date:_____

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in US$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.2.2 Irie Pascal (Linux Edition) CA$ Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	99.99	_____	$_____
5	87.49	_____	$_____
10	74.99	_____	$_____
20	62.49	_____	$_____
40	49.99	_____	$_____
100	37.49	_____	$_____
Shipping & Handling	7.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in CA$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.2.3 Irie Pascal (Linux Edition) UK Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	39.99	_____	_____
5	34.99	_____	_____
10	29.99	_____	_____
20	24.99	_____	_____
40	19.99	_____	_____
100	14.99	_____	_____
Shipping & Handling	3.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in UK£ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be*

notified by email when your order is shipped.

5.7.9.2.4 Irie Pascal (Linux Edition) Euro Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	59.99	_____	_____
5	52.99	_____	_____
10	45.99	_____	_____
20	38.99	_____	_____
40	31.99	_____	_____
100	24.99	_____	_____
Shipping & Handling	4.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in Euros and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.3.1 Irie Pascal (FreeBSD Edition) US$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	79.99	_____	$_____
5	69.99	_____	$_____
10	59.99	_____	$_____
20	49.99	_____	$_____
40	39.99	_____	$_____
100	29.99	_____	$_____
Shipping & Handling	5.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed [ ] Charge credit card

[ ] AmEx [ ] MasterCard [ ] Visa

Account #:_____

Expiration Date:_____

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in US$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.3.2 Irie Pascal (FreeBSD Edition) CA$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	99.99	_____	$_____
5	87.49	_____	$_____
10	74.99	_____	$_____
20	62.49	_____	$_____
40	49.99	_____	$_____
100	37.49	_____	$_____
Shipping & Handling	7.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in CA$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.3.3 Irie Pascal (FreeBSD Edition) UK Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	39.99	_____	_____
5	34.99	_____	_____
10	29.99	_____	_____
20	24.99	_____	_____
40	19.99	_____	_____
100	14.99	_____	_____
Shipping & Handling	3.00		_____

No. Users	Unit Price	Quantity	Total Price

TOTAL _____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in UK£ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.3.4 Irie Pascal (FreeBSD Edition) Euro Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	59.99	_____	_____
5	52.99	_____	_____
10	45.99	_____	_____
20	38.99	_____	_____
40	31.99	_____	_____
100	24.99	_____	_____

No. Users	Unit Price	Quantity	Total Price
Shipping & Handling	4.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in Euros and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.4.1 Irie Pascal (Solaris/x86 Edition) US$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	79.99	_____	$_____
5	69.99	_____	$_____
10	59.99	_____	$_____
20	49.99	_____	$_____

No. Users	Unit Price	Quantity	Total Price
40	39.99	_____	$_____
100	29.99	_____	$_____
Shipping & Handling	5.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed [ ] Charge credit card

[ ] AmEx [ ] MasterCard [ ] Visa

Account #:_____

Expiration Date:_____

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in US$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.4.2 Irie Pascal (Solaris/x86 Edition) CA$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	99.99	_____	$_____
5	87.49	_____	$_____
10	74.99	_____	$_____
20	62.49	_____	$_____
40	49.99	_____	$_____
100	37.49	_____	$_____
Shipping & Handling	7.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in CA$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.4.3 Irie Pascal (Solaris/x86 Edition) UK Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	39.99	_____	_____
5	34.99	_____	_____
10	29.99	_____	_____
20	24.99	_____	_____
40	19.99	_____	_____
100	14.99	_____	_____
Shipping & Handling	3.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in UK£ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be*

notified by email when your order is shipped.

5.7.9.4.4 Irie Pascal (Solaris/x86 Edition) Euro Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	59.99	_____	_____
5	52.99	_____	_____
10	45.99	_____	_____
20	38.99	_____	_____
40	31.99	_____	_____
100	24.99	_____	_____
Shipping & Handling	4.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in Euros and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.5.1 Irie Pascal (Solaris/Sparc Edition) US$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	79.99	_____	$_____
5	69.99	_____	$_____
10	59.99	_____	$_____
20	49.99	_____	$_____
40	39.99	_____	$_____
100	29.99	_____	$_____
Shipping & Handling	5.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed [ ] Charge credit card

[ ] AmEx [ ] MasterCard [ ] Visa

Account #:_____

Expiration Date:_____

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in US$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.5.2 Irie Pascal (Solaris/Sparc Edition) CA$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	99.99	_____	$_____
5	87.49	_____	$_____
10	74.99	_____	$_____
20	62.49	_____	$_____
40	49.99	_____	$_____
100	37.49	_____	$_____
Shipping & Handling	7.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in CA$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.5.3 Irie Pascal (Solaris/Sparc Edition) UK Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	39.99	_____	_____
5	34.99	_____	_____
10	29.99	_____	_____
20	24.99	_____	_____
40	19.99	_____	_____
100	14.99	_____	_____
Shipping & Handling	3.00		_____

No. Users	Unit Price	Quantity	Total Price

TOTAL _____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in UK£ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.5.4 Irie Pascal (Solaris/Sparc Edition) Euro Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	59.99	_____	_____
5	52.99	_____	_____
10	45.99	_____	_____
20	38.99	_____	_____
40	31.99	_____	_____
100	24.99	_____	_____

No. Users	Unit Price	Quantity	Total Price
Shipping & Handling	4.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in Euros and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.6.1 Irie Pascal (Universal Edition) US$ Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	149.99	_____	$_____
5	131.99	_____	$_____
10	113.99	_____	$_____
20	95.99	_____	$_____

No. Users	Unit Price	Quantity	Total Price
40	77.99	_____	$_____
100	59.99	_____	$_____
Shipping & Handling	5.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed [ ] Charge credit card

[ ] AmEx [ ] MasterCard [ ] Visa

Account #:_____

Expiration Date:_____

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in US$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.6.2 Irie Pascal (Universal Edition) CA$ Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	187.49	_____	$_____
5	164.99	_____	$_____
10	142.49	_____	$_____
20	119.99	_____	$_____
40	97.49	_____	$_____
100	74.99	_____	$_____
Shipping & Handling	7.00		$_____
TOTAL			$_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in CA$ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

5.7.9.6.3 Irie Pascal (Universal Edition) UK Order Form

IRIE TOOLS

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	74.99	_____	_____
5	65.99	_____	_____
10	56.99	_____	_____
20	47.99	_____	_____
40	38.99	_____	_____
100	29.99	_____	_____
Shipping & Handling	3.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

[ ] Tick this box to join the Irie Pascal mailing list.

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in UK£ and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be*

notified by email when your order is shipped.

5.7.9.6.4 Irie Pascal (Universal Edition) Euro Order Form

Irie Tools
221 S. State Rd 7, #247
Ft. Lauderdale, FL 33317, USA

web: www.irietools.com
Email: sales@irietools.com

No. Users	Unit Price	Quantity	Total Price
1	112.49	_____	_____
5	98.99	_____	_____
10	85.49	_____	_____
20	71.99	_____	_____
40	58.49	_____	_____
100	44.99	_____	_____
Shipping & Handling	4.00		_____
TOTAL			_____

[ ] Check/Money Order enclosed

Name:_____

Address:_____

City/State/Province:_____

Country/Postal Code:_____

Telephone:_____ [optional]

Email Address:_____ [optional]

How did you hear about Irie Pascal?_____

_____ [optional]

```
[  ] Tick this box to join the Irie Pascal mailing list.
```

*Please do **not** send cash. Make checks or money orders payable to **Irie Tools**. All prices are quoted in Euros and are guaranteed until December 31, 2005. Please allow two to four weeks for delivery. If you fill in your email address you will be notified by email when your order is shipped.*

" Assignment overflow"

When this option is enabled, your program will check each time a value is assigned to a string or set variable, to make sure that the value is not too long to fit. If the value it too long to fit in the variable then an assignment overflow error is generated.

" I/O errors"

When this option is enabled, your program will check each I/O operation for errors.

" Range errors"

When this option is enabled, your program will check for range errors. Some possible causes of range errors are:

- Attempts to assign or read in values which are not assignment compatible with a particular type to a variable of that type. For example given the declaration below:

  ```
  var bit : 0..1;
  ```

  the assignment below will cause a range error:

  ```
  bit := 2;
  ```

- Attempts to access element outside of array bounds. For example given the declaration below:

  ```
  var x : array[-4..10] of integer;
  ```

  the attempt below to reference an out-of bounds array element will cause a range error:

  ```
  x[-5]
  ```

" Stack overflow"

When this option is enabled, your program checks for stack overflow or underflow before each program statement is executed. When this option is not enabled, your program checks for stack overflow or underflow only at the beginning and end of each function/procedure call and when large values are placed on the stack.

" Using undefined values"

When this option is enabled, your program checks each time a value, from a variable, or returned by a function call, is accessed, to make sure that the value is not undefined. **NOTE:** Not all values can be checked, checks are only made for values of the following types:

- enumerated types (including boolean)
- subranges of enumerated types

- subranges of integer that do not include -1
- file types
- list types
- object types
- pointer types
- real
- set types (array representation)

Accesses to values of types: **char**, **integer**, **record types** and **set types (Bit set representation)** are not checked.

" Using in-active variants"

This check box controls whether your program checks each time a field of a variant is accessed, to make sure that the variant is active.

" On the console screen"

This check box controls whether run-time errors are displayed on your program's console window.

" in message boxes"

This check box controls whether run-time errors are displayed in Windows' message boxes.

" in log files"

This check box controls whether run-time errors are logged to a file named **name.log** (where **name** is the name of your program).

" Enable asserts"

This check box controls whether the compiler generates code for the built-in procedure **assert**. **NOTE**: Since the compiler parses **assert** procedures whether or not this check box is checked, compile-time errors in **assert** procedures are reported regardless of the setting of this check box.

" Generate Windows NT/2000 service application"

Check this box to compile your program as a Windows NT/2000 service. **NOTE:** Your program must be a EXE executable.

" Use short-circuit evaluation"

This check box controls whether your program uses short-circuit evaluation for the boolean operators **and** and **or**. When short-circuit evaluation is used for the boolean **and** operator, your program evaluates the left-hand operand and if the result is **false**, then the right-hand operand is not evaluated because the result of the operation must be **false**. When short-circuit evaluation is used for the boolean **or** operator, your program evaluates the left-hand operand and if the result is **true**, then the right-hand operand is not evaluated because the result of the operation must be **true**.

You might want to disable short-circuit evaluation if you need the side-effects of evaluating the right

operand. For example, suppose the right operand is a call to a function which modifies some global variables (a side-effect) in addition to returning a boolean value, then short-circuit evaluation might cause the function not to be called and therefore the global variables will not get modified. If you want to make sure that the right operand is always evaluated then disable short-circuit evaluation.

" Insert line-number debugging information"

This check box controls whether line-number debugging information is inserted into your program. Line number debugging information makes your program larger, but allows for more meaningful run-time error messages since they will include number of the source line that caused the error.

" Specify align size"

This allows you to control the maximum alignment used by the compiler. Some CPUs (including those in the Intel 80x86 family) access data faster if it is aligned on an address which is a multiple of the size of the data. Suppose the CPU is accessing a real which is 8 bytes long, then for fastest access, the real should be on an address which is a multiple of 8 (for example 0, 8, 16, 24, 32, etc).

The compiler stores variables in memory at the lowest available address which is a multiple of either the variable's size or the maximum alignment, whichever is smaller.

For example suppose you compile the following program and the maximum alignment is 4.

```
program x(output);
var
c : char;
i : integer;
b : boolean;
r : real;
begin
end.
```

The compiler needs to decide where to store the variables **c**, **i**, **b** and **r**. The first variable **c** gets stored at address 0, and since **c** is a char variable (which are 1 byte long) the available addresses are from 1 upwards. The second variable **i** is an integer variable (which are 4 bytes long). The lowest available address which is a multiple of the variable size is 4 and the lowest available address which is a multiple of the maximum alignment is also 4. So the compiler stores **i** at address 4, and the available address are from 8 upwards (since **i** is 4 bytes long). The third variable **b** is a boolean variable (which are 4 bytes long). The lowest available address which is a multiple of the variable size is 8 and the lowest available address which is a multiple of the maximum alignment is also 8. So the compiler stores **b** at address 8, and the available address are from 12 upwards (since **b** is 4 bytes long). The fourth variable **r** is a real variable (which are 8 bytes long). The lowest available address which is a multiple of the variable size is 16 but the lowest available address which is a multiple of the maximum alignment is 12. So the compiler stores **r** at address 12 (since 12 is less than 16).

In general if you set maximum alignment to 1 then you waste no memory but you get fastest access only for chars. If you set the maximum alignment to 4 you may waste memory when storing all variables except char, but you get the fastest access to all variables except real. If you set the maximum alignment to 8 you may waste memory when storing all variables except char, but you get the fastest access to variables of all types.

If you are not sure about the maximum alignment to use just leave the default (4).

" Specify stack size"

This allows you to control the size of your program's stack (in Kilobytes). The default stack size (64K) should be more than enough for the vast majority of programs. However if your program is heavily recursive or has functions that need a large amount of space for local variables or parameters then you can increase this value up to a maximum of 1024K (1MB). On the other hand you can also reduce the size of your program's stack (not recommended).

" Use tab character"

When this check box is checked the editor will use the tab character to store tabs. If tab characters are used then you can adjust the size of the tabs using the **tab size** checkbox.

When this check box is not checked, the editor will convert tabs to spaces before storing them. Tabs are converted to spaces whether they are entered using the keyboard, or occur in files opened by the editor.

" Auto-indent lines"

This check box controls whether the editor will automatically indent new lines to match the indentation of the previous line.

" Create backup files"

This check box controls whether the editor will create a backup when files are saved. The backup file has the same name as the original file, except with the extension (.bak). If the original file already has a (.bak) extension then no backup is created.

" Double-click performs word search"

When this check box is checked double-clicking on a word in the editor will cause the IDE to search the Irie Pascal Programmer's Reference Manual (in "progref.html") for help on that word.

When this check box is not checked double-clicking on a word in the editor will cause the editor to select the word.

" Tab size"

This allows you to specify the number of characters between tab stops. The default tab size is 8. If you use tabs to indent your programs you might want to reduce this size so that more of your code fits on the screen.

" Name of your program's executable"

If you want to change the name of the executable generated by the compiler, enter the new name here. By default, the name of the executable generated by the compiler will be the same as the name of the project, except with the extension **.ivm**. If you enter a name with the extension **.exe**, the compiler will generate a true EXE executable. If the name here does not have an extension or has an extension other than **.exe** then the compiler will generate an IVM executable.

" Arguments passed by the IDE"

This is where you would enter any arguments you want to pass to your program when it is run from inside the IDE.

" #! header"

This is where you would enter any #! header that you want in the executable generated by the compiler. UNIX like operating systems such as Linux, FreeBSD, and Solaris will use the #! header to locate the interpreter. This allows you to run your executable without specifying the interpreter (a necessity for CGI applications). NOTE: You also need to set the executable permission bit of the executable for this to work. IMPORTANT: This header is ignored under non-UNIX like operating systems, so including this header does not prevent the executable from running under any supported operating system.

For example, if you plan to deploy your executable on Linux, and you know that the full pathname of the interpreter is **/usr/local/bin/ivm**, then you would just enter that pathname into this text box, and the compiler will generate the #! header for you.

NOTE: The command-line compiler can also generate #! headers (see the -h compiler option). If you wish to add, change, or delete the #! header of an existing executable then you can use the Irie Header Utility (see using the Irie Header Utility). Which can be useful if you can't, or don't want to recompile the program.

" Allow nested comments"

This check box controls whether nested comments (which are comments inside other comments) are supported. For example

```
(* outer (* inner comment *) comment *)
```

When nested comments are not supported the example comment above will terminate at the first ***)** so only

```
(* outer (* inner comment *)
```

will be treated as a comment. When nested comments are supported the compiler recognizes the end of comments only when the number of close comment markers matches the number of open comment markers. So the example comment above will terminate only after the second ***)**.

Both open comment markers **(*** and **{** are considered to be equivalent, and both close comment markers ***)** and **}** are considered to be equivalent. So attempting to trick the compiler into accepting nested comments with something like

```
(* outer { inner comment } comment *)
```

will not work.

Nested comments are disabled by default since in Standard Pascal comments do not nest.

" Make identifiers case-sensitive"

When this check box is checked identifiers are case-sensitive, so for example the following are all different identifier (**hi**, **Hi**, **hI**, and **HI**).

When this check box is not checked, identifiers are not case-sensitive, so the identifiers in the previous example are all the same identifier.

Case-sensitive identifiers are disabled by default, since Pascal is normally not a case-sensitive language. If you enable this option then remember to use all lowercase for keywords and built-in identifiers (such as **var**, **integer**, **input**, **writeln**).

" Require parentheses"

This check box controls whether parentheses are mandatory in all function and procedure calls and after the program name. In Pascal parentheses are not normally used when calling or declaring functions or procedures with no parameters. So for example in the following statement:

```
a := b;
```

it is not clear whether **b** is a function that takes no parameters or whether **b** is a variable. When mandatory parentheses mode is enabled then parentheses are required when declaring or calling all functions and procedures even those with no parameters. Parentheses are also required after the program name even if there are no program parameters.

" Non-standard unary operators"

This check box controls whether the compiler handles the unary plus and unary minus operators in a non-standard way (i.e. different from the way specified by Standard Pascal). The compiler will allow the unary plus and unary minus operators to appear before any numeric operand, and their precedence is higher than any other operator.

" Open temp file if no name assigned"

Normally, if you open a file variable, of text type, without first assigning it a name, then the file variable will become associated with the the standard input stream or the standard output stream, depending on whether you are opening the file variable for reading or writing. In which case reading from the file variable will read from the standard input stream, and writing to the file variable will write to the standard output stream.

When this check box is checked and you open a file variable, of text type, without first assigning it a name, then the file variable will become associated with a temporary disk file and not the standard input stream or the standard output stream.

NOTE: Opening a file variable, that is **not** of text type, without first assigning it a name, will always associated the file variable with a temporary disk file, regardless of whether this check box is checked.

" Compatibility mode (with version 2.0)"

This check box controls whether the compiler is in version 2.0 compatibility mode. When you compile your program with the compiler in this mode, your program should behave in the same way it did when compiled with verions 2.0 or 2.1 of the compiler. Compatibility with versions earlier than 2.0 is not guaranteed.

In this version of the compiler the only effect of version 2.0 compatibility mode, is to make the compiler continue to incorrectly treat sets of subrange types as if they were sets of the subrange's host type. So for example

```
set of 0..1
```

is treated like

```
set of integer
```

which in this case will affect how values of the set are represented. Normally your program is not affected by the way set values are represented, however if your program stores set values in data files then the representation of the set values is important.

Compatibility mode was introduced so that you can maintain the format of data files created by programs compiled with previous versions of the compiler, even when you compile those programs with this version of the compiler.

" Maximum number of errors allowed"

This is where you would enter the maximum number of errors you will allow the compiler to report before it should give up and stop compiling.

" Maximum number of warnings allowed"

This is where you would enter the maximum number of warning you will allow the compiler to report before it should give up and stop compiling.

Access keys are associated with some menus and menu items, and are indicated by underlined letters in the menu titles and the text of the menu items.

A *compiler* translates programs written in one programming language into programs written in another programming language. The program being translated is usually written in a high-level language (like Pascal), and the program that results from the translation is usually written in a low-level language. This process of translating programs from one programming language to another is called *compiling*, and the program that performs this process is called a *compiler*.

Compiling is the process of translating programs written in one programming language into equivalent programs written in another programming language. The program being translated is usually written in a high-level language (like Pascal), and the program that results from the translation is usually written in a low-level language. The program that does the *compiling* is called a *compiler*.