Zonnon Builder User Guide

1. Introduction

The **Zonnon Builder** is a .NET based IDE (Integrated Development Environment) designed to facilitate the process of developing and managing programs in the **Zonnon** language which run on the **.NET platform**.

It supports the notion of a **Project** as a repository used to hold all the folders and files for a program, or if desired for several related programs.

A **syntax oriented editor** is used for viewing and editing **Zonnon source files.**

The **search engine** can be used within a single source file or across a selected set of files in a project making it convenient to track the implications of proposed changes across the project .

A **version management system** is included to control the **revisions** and **versions** of sets of files that form consistent software releases within a project.

Of course the heart of the Builder is the **Zonnon Compiler** for .NET which rapidly compiles Zonnon source files ready for execution on the **.NET platform**.

Note that the **Zonnon Builder** runs on top of the Microsoft .NET platform, this needs to be installed *before* installing and running the Zonnon Builder.

Zonnon Builder User Guide

2. Zonnon Builder Installation

If you don't already have the **.NET Framework** installed then the first thing to do is to download it from the Microsoft site and install it. Now just follow the following four steps to install **Zonnon Builder** ...

Step 1: Unzip the **ZonnonBuilder.zip** distribution file to the directory of your choice.

Step 2: Execute the **Setup.exe** program. The program will install Zonnon Builder to the directory **\Program Files\ETH Zurich\ZonnonBuilder**.

Step 3: Finally you need to execute the program **ethz.zonnon.builder.exe** to start the **Zonnon Builder**. The **Zonnon Builder** icon on decktop or in main **Start** menu may be used also for the execution.

Figure 2.1. The **Zonnon Builder** icon on desktop.

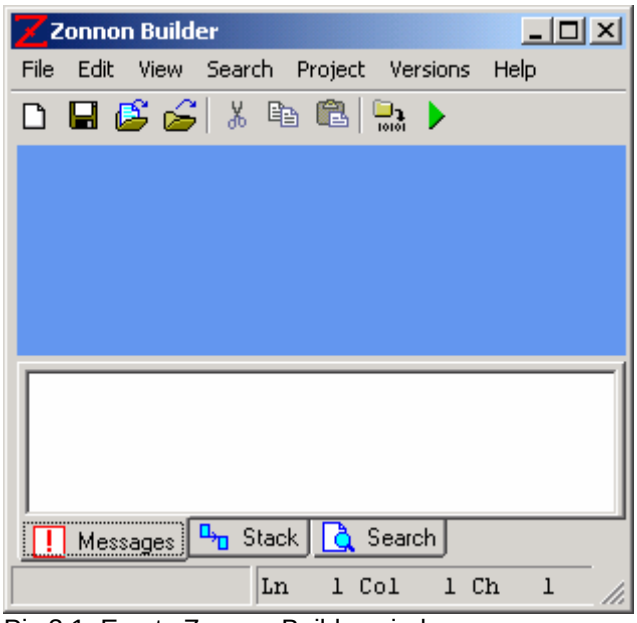
Figure 2.2. The **Zonnon Builder** icon in main **Start** menu.

Enjoy working with the **Zonnon Builder** and developing programs in **Zonnon** for the .NET platform.

3. Getting started

3.1. The main window structure

When Zonnon Builder starts up you will see the initial empty window :



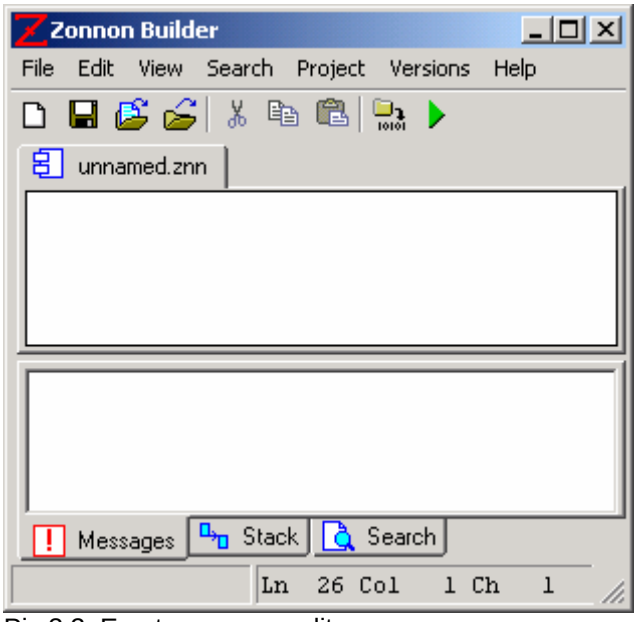
Pic.3.1. Empty Zonnon Builder window.

The central window panel will contains program file editors and file history viewers.

The bottom window panel has tabs for **Messages**, **Stack** and **Search**. The **Messages** tab is used to show messages from Zonnon Compiler. The **Stack** tab is used to show the Zonnon program stack at the point when program execution was last terminated. The **Search** tab is used to show text search results.

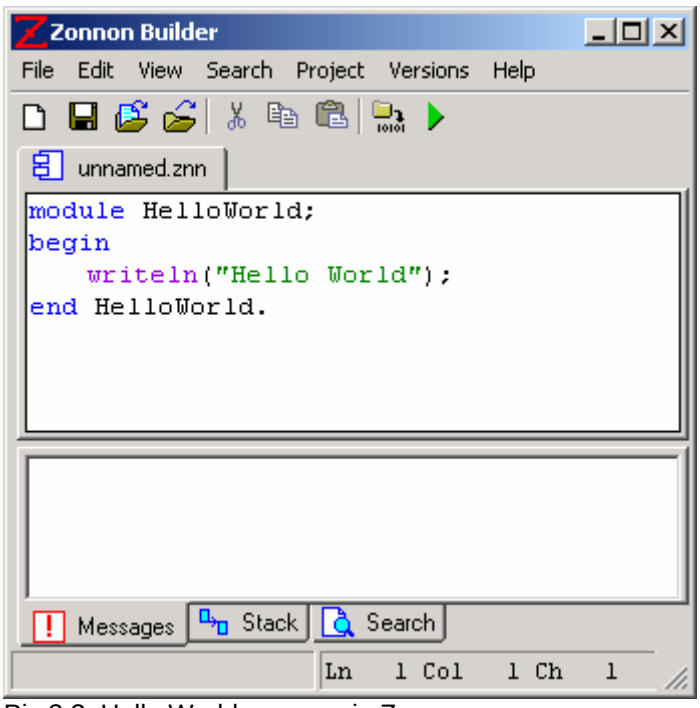
3.2. Zonnon program from file

To create a first Zonnon program click the **New** icon in main toolbar. The empty program editor will be shown at the central window panel:



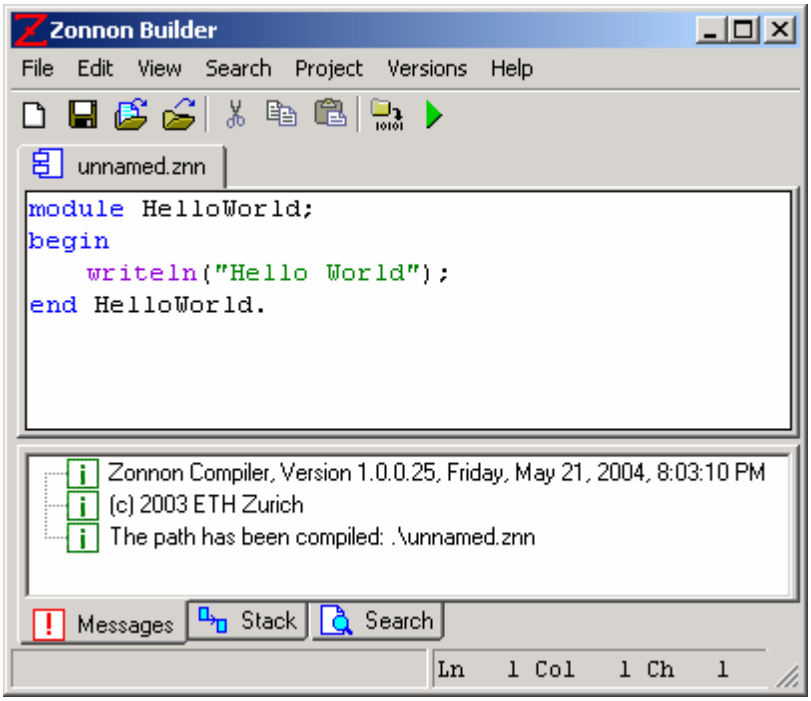
Pic.3.2. Empty program editor.

Lets input simples HelloWorld program in Zonnon language:



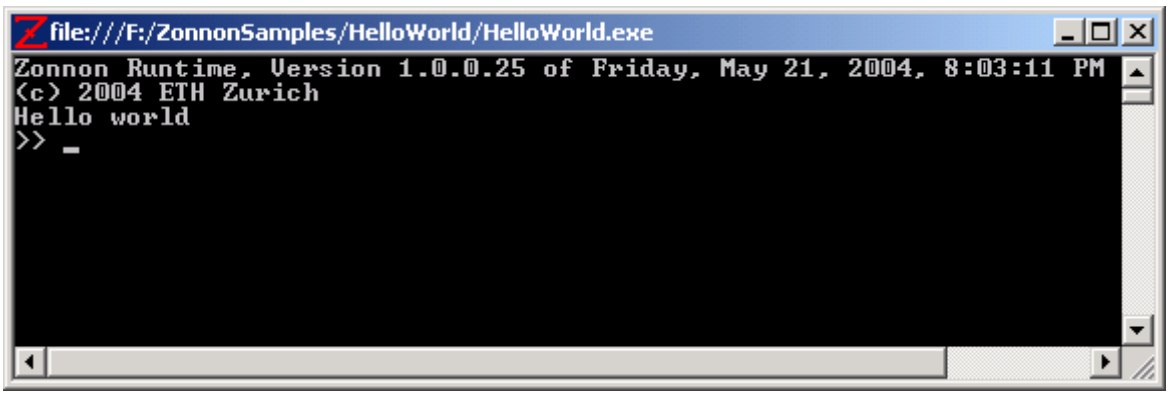
Pic.3.3. Hello World program in Zonnon.

To compile program press the keys combination **Ctrl+Shift+F9**. The compilation results will be presented at the panel **Messages**:



Pic.3.4. HelloWorld program compilation results.

The executable file **HelloWorld.exe** was created at same directory where the file **HelloWorld.znn** is located. To execute the compiled **HelloWorld** program press the key **F9**. The console window with the messages "Hello World" will be created:



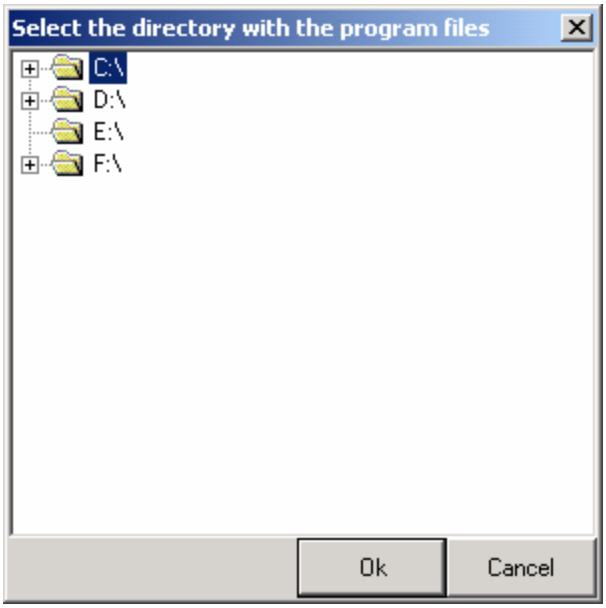
Pic.3.5. HelloWorld program execution results.

Then Zonnon Runtime environment output the prompt >> for a command input. Now print the command **quit** and press the **Enter** key to out from Zonnon Runtime program.

The command **help** can be used to receive information about other commands. The command **list** shows all program modules and its procedures. To start a procedure print the module and procedure names and press the **Enter** key.

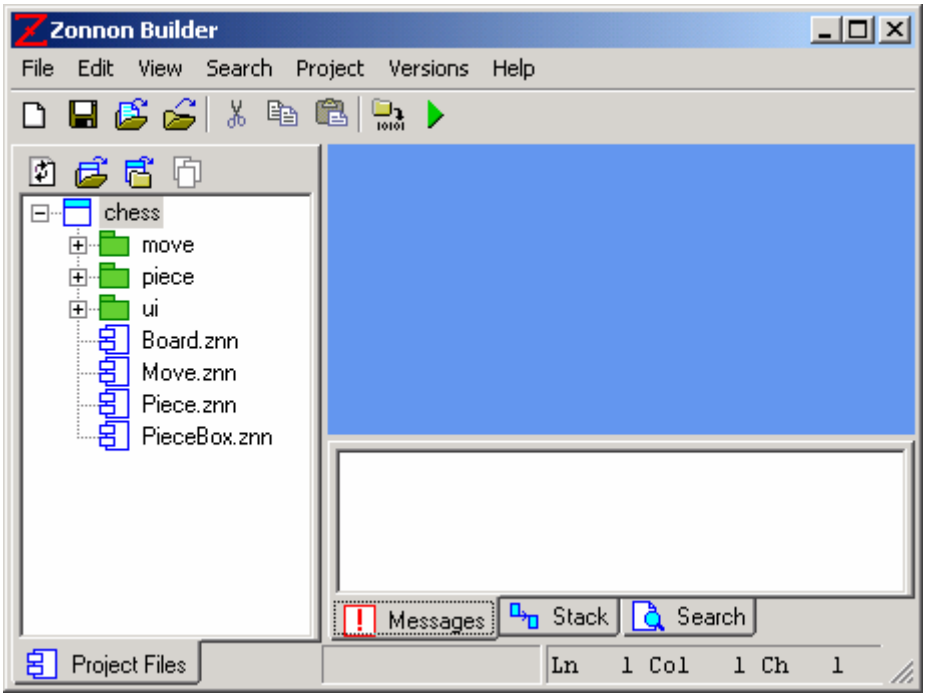
3.3. Zonnon program with few files

More complicated Zonnon program contains few program files. To use the files in Zonon Builder its must be located in same root directories or in nested directories. For example, the chess program files are located in the root directory chess or in nested directories. To open the program files select the menu item **File | Open Directory**. The following dialog box appears:



Pic.3.6. The directory selection dialog box.

Select some directory with Zonnon programs and press the **Ok** button. The program tree with the program files will appears in the left panel of main window:



Pic.3.7. The program tree.

To compile all program files press the keys combination **Ctrl+F9**. The compilation results will be presented at the panel **Messages**. The executable file name has the root name and the extention exe: **chess.exe**. It is located in the chess directory. The program directory may be interpreted as *project* with default properties.

To change executable file name and location, to set the program main module, see Project Setting for details.

4 Editing a Program

4.1 File opening and closing.

To edit any Zonnon program file select the menu item **File | Open File...** in the main Zonnon Builder menu:

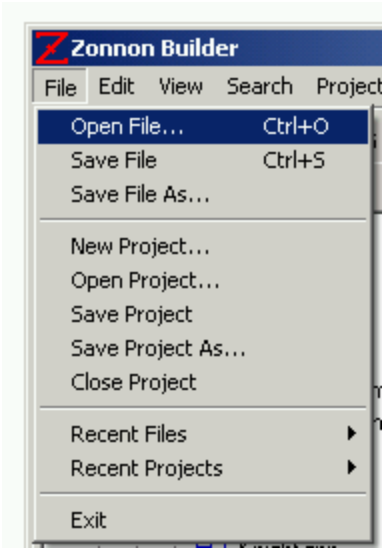


Figure 4.1 The **Open File** menu item in the main menu.

Alternatively, right click on the **Open File** icon in main toolbox :

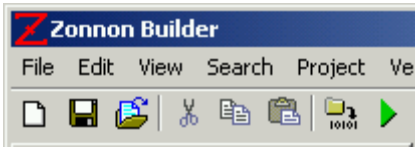


Figure 4.2 **Open File** icon in main toolbar.

It is also possible to open any Zonnon program file by double clicking on it in the program tree and then using the **Open** popup menu item that then appears :

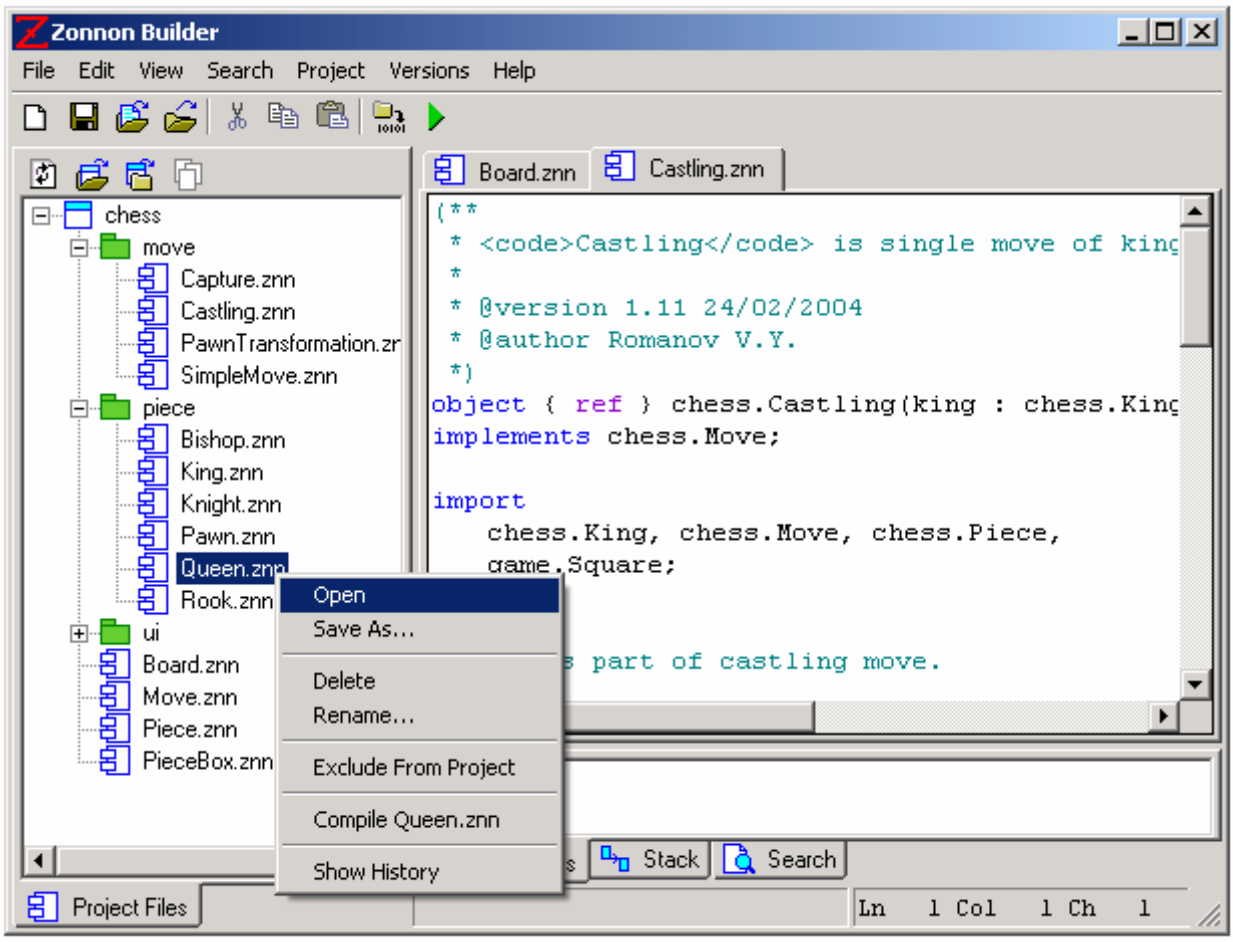


Figure 4.3 Popup menu of program tree file item.

Now lets open a few Zonnon program files in the **Zonnon Chess** project :

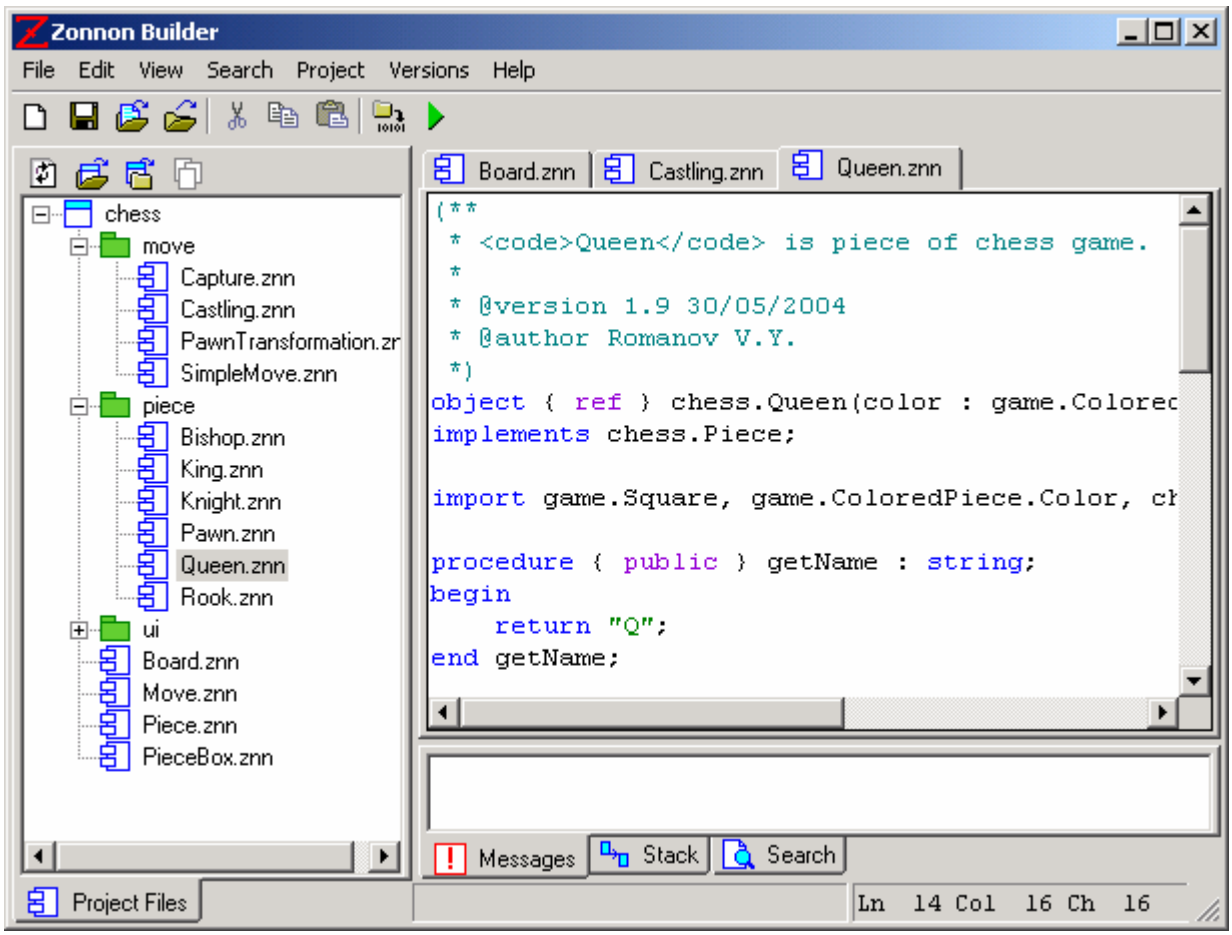


Figure 4.4 The Zonnon program file editor.

Note that different syntactic elements of the program source text are highlighted in different colours to improve clarity. Zonnon Builder updates this syntax oriented program text colouring after each key stroke is entered.

4.2 Editor popup menu.

Each file has an associated tab with a popup menu :

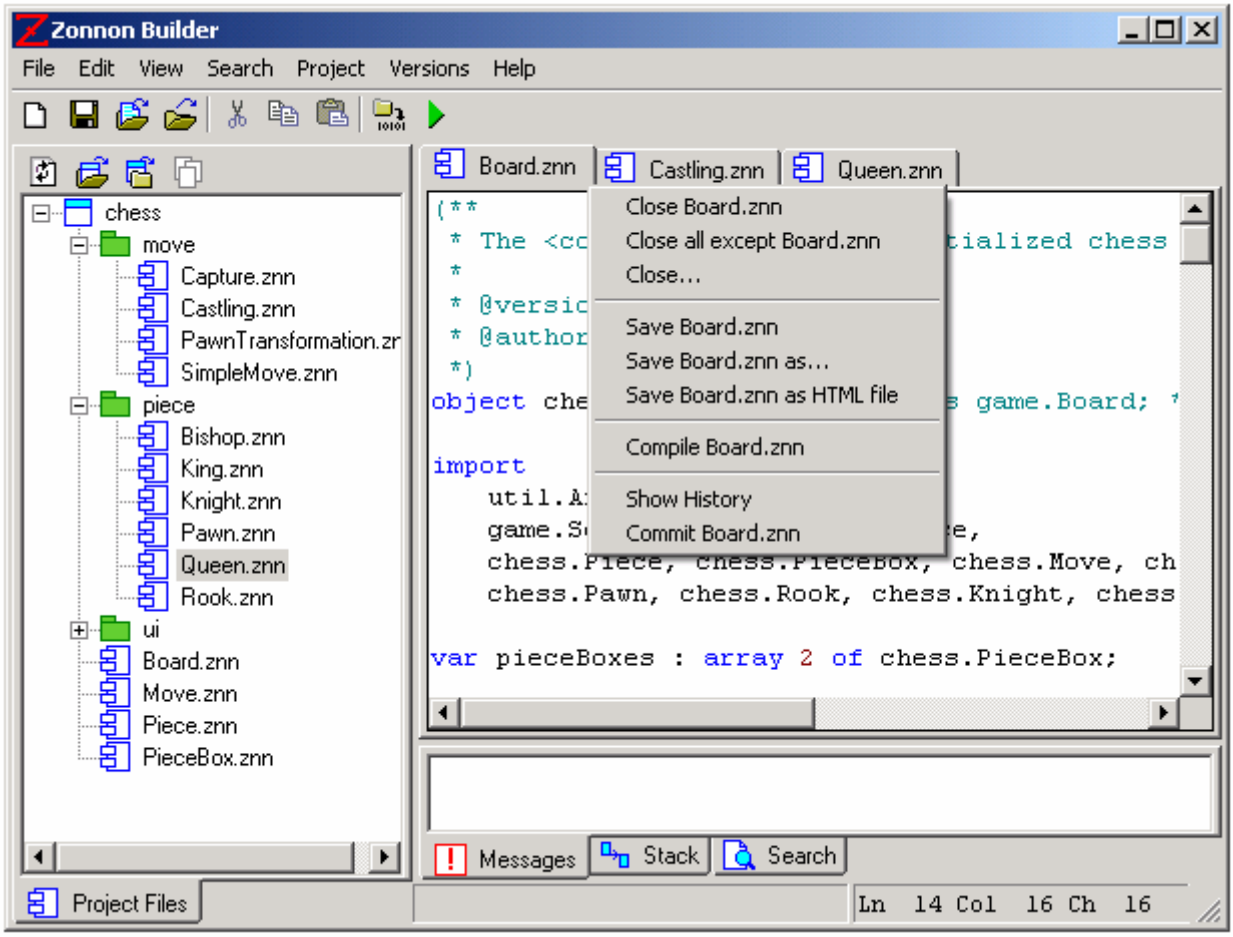


Figure 4.5 Program editor popup menu.

Closing files is quite flexible. It is possible to close either the selected file **Castling.znn** or close all files *except* the selected file **Castling.znn** by using the menu items **Close** or **Close all except**. Any group of opened files can be closed using the menu item **Close...** , when it is clicked the following dialog box appears:

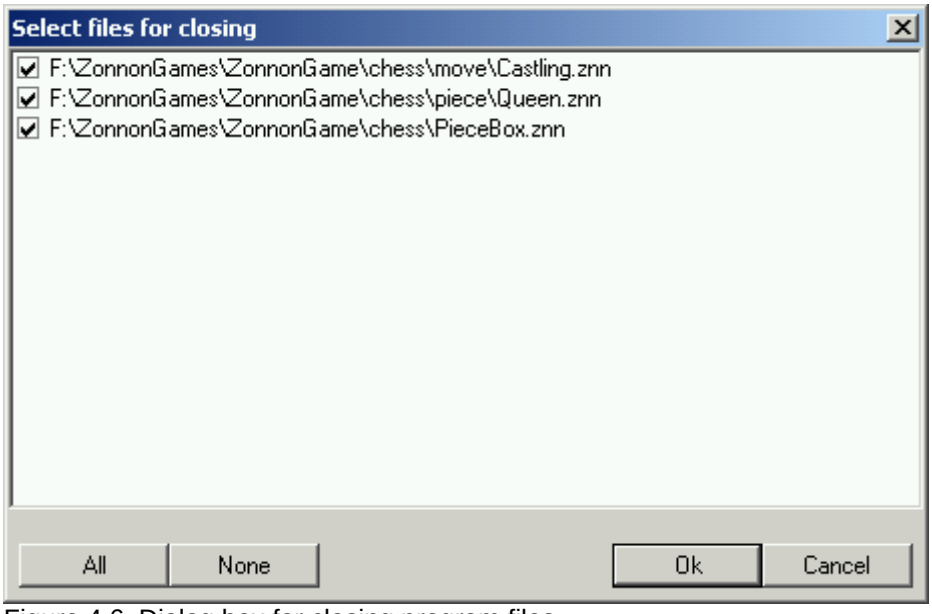


Figure 4.6 Dialog box for closing program files .

The selected file can be saved with either the same name or with a different name. The menu items **Save Castling.znn** or **Save Castling.znn as...** are used for this. If you want to publish your Zonnon program in

an Internet based format then use the menu item **Save Castling.znn as HTML**. The HTML file **Castling.znn.html** will contain your Zonnon program, it is coloured to highlight the program's syntax.

The Zonnon source text can be compiled to create a program (or part of a program) by clicking the **Save Castling.znn** menu item. For further details see the chapter on **Program Compilation** .

The history of the Zonnon program file can be viewed by clicking on the **Show History** menu item. A History Window will opened for the file, it contains the :

- content and modification date for each file revision and date.
- the revision, version and current state of the file, using colour coding.
- comments associated with each file version.

If the file is now completed it can be saved as the next commented *version* of the file by clicking on the menu item **Commit Castling.znn**. For further details see the chapter on **Versioning**.

To edit the actual file content itself then it is possible to use the menu **Edit** in the main Zonnon Builder menu:

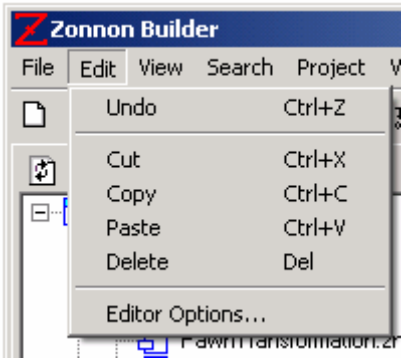


Figure 4.7 **Edit** menu in main Zonnon Builder menu.

Also you can use icons **Cut** , **Copy** and **Paste** in the main toolbar:

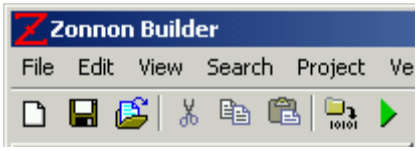


Figure 4.8 Main toolbar.

- the **Cut** icon - the **Copy** icon - the **Paste** icon

It is possible to customise the 'look' of the program text, see **Editor Options** for details.

5. Searching Text in Program Files

5.1 Searching for text in all program files.

Sometimes it is desirable to find a particular piece of text in all the files in a program. To achieve this select the menu item **Find in Project** in the **Search** main menu :

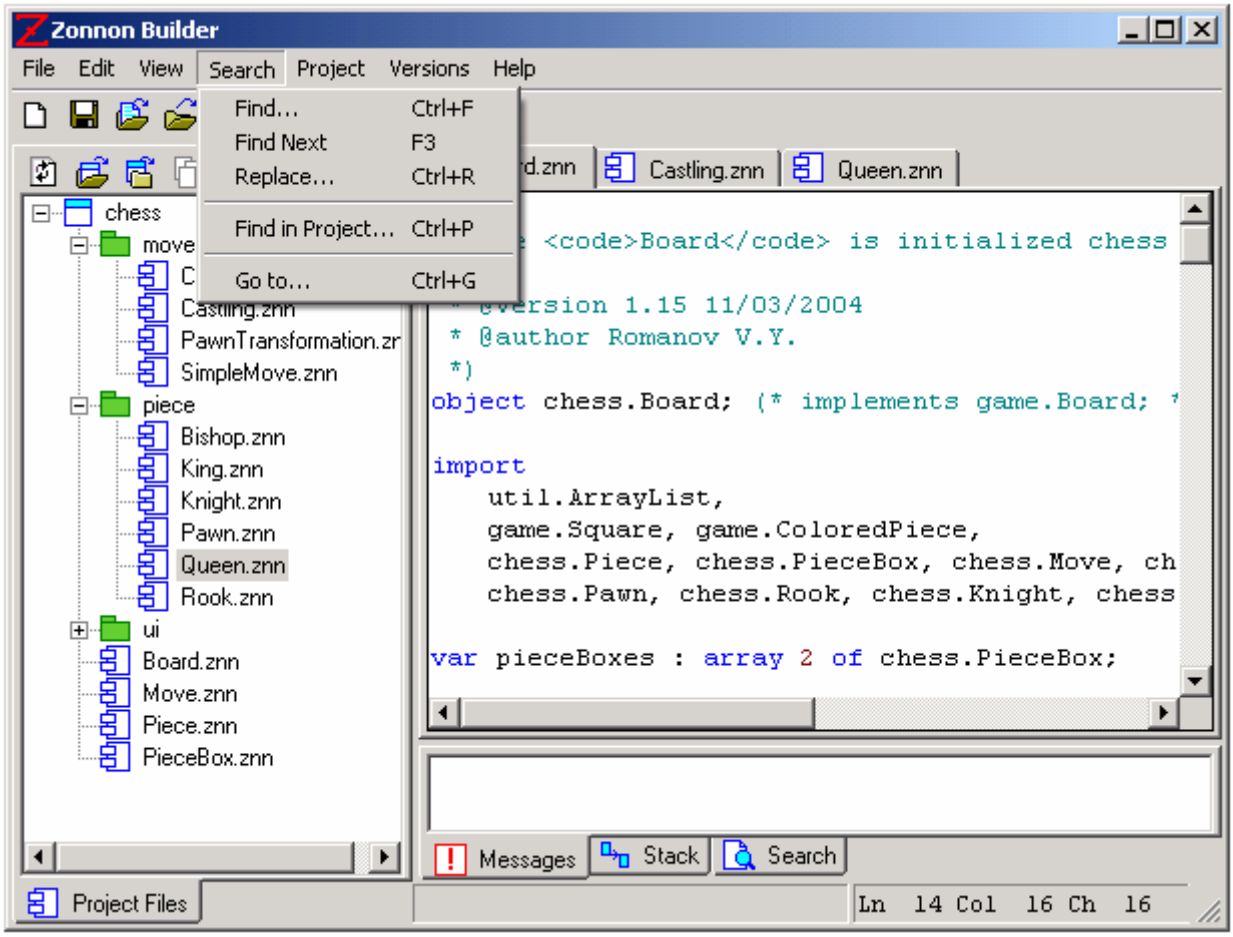


Figure 5.1 The Zonnon Builder **Search** menu.

The following dialog box will appear :

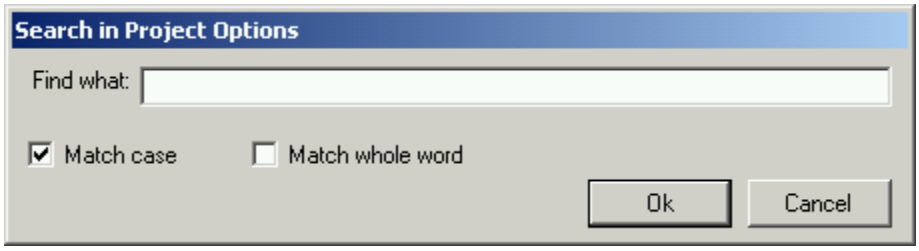


Figure 5.2 **Search in Project Options** dialog box.

Lets look for the program text files with the word **Pawn**. Type the text to be searched for in the **Find What** textbox and then press **Ok** button. The search results will presented in the **Search** tab of the bottom window pane on the right :

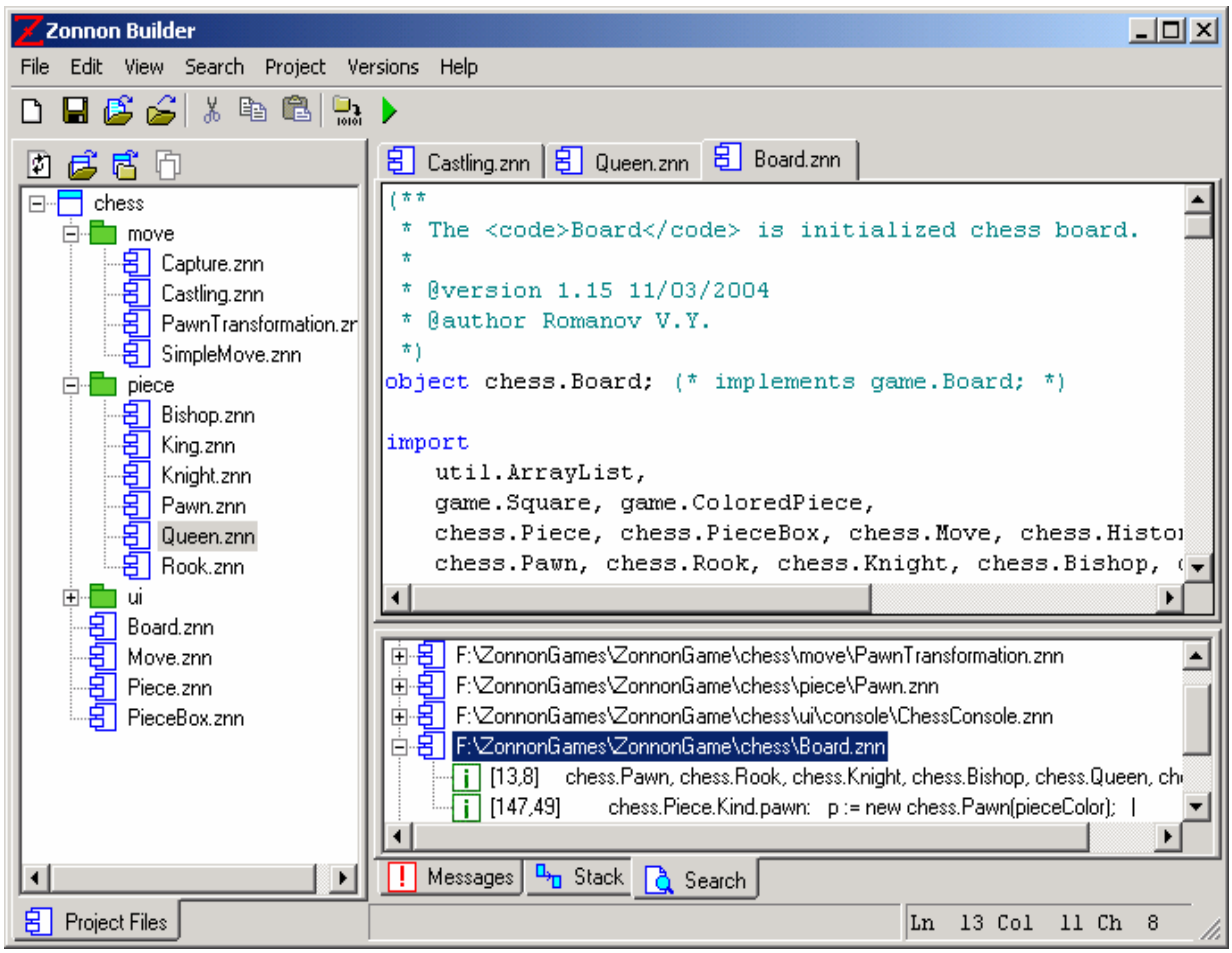


Figure 5.3 The **Search** tab with search results.

This shows that six files contain the word **Pawn** and it was found a total of 24 times. Lets open the search result in file **Board.znn**, it contains two strings that include the word **Pawn**. Line number 13 contains **Pawn** in column 8 and line 144 contains **Pawn** in column 49. Double clicking on a line opens the **Board.znn** file, the line and column location is highlighted in red :

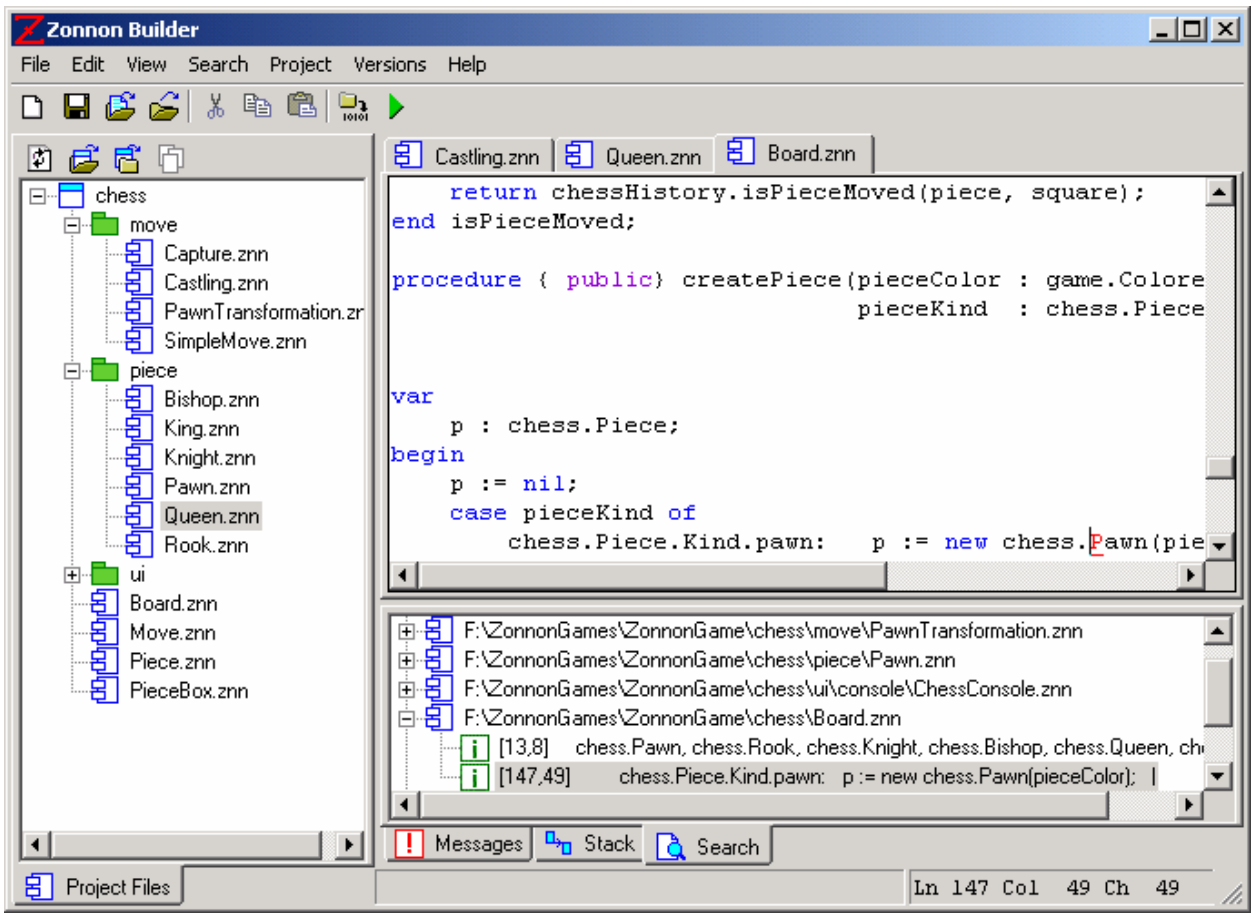


Figure 5 .4 Search result colouring in the Zonnon Editor.

5.2 Searching for text in a file.

To search within a file that is already open use **Search | Find from** the main menu item. The following dialog box will appear :

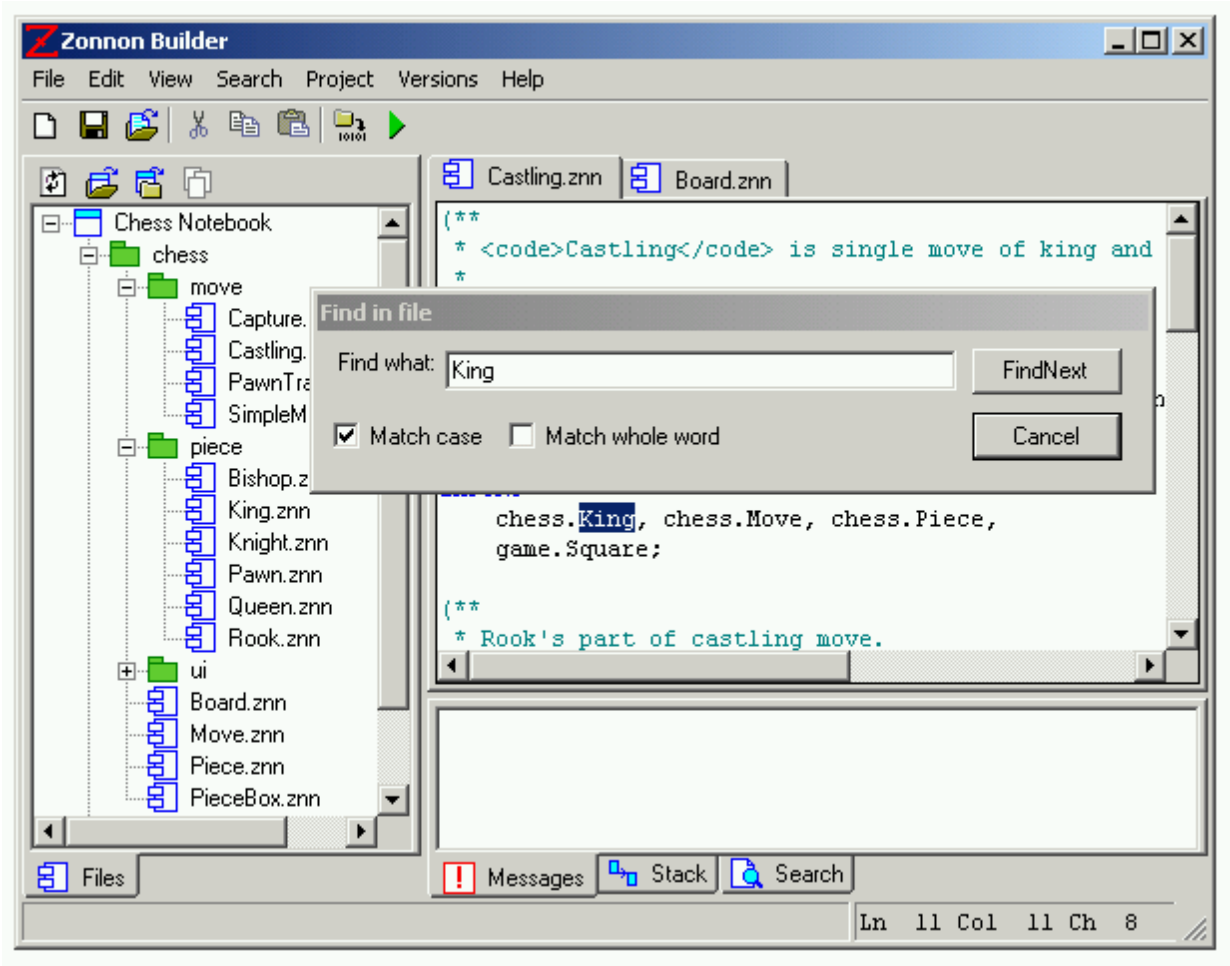


Figure 5.5 The **Find Options** dialog.

Enter the piece of text to be searched for in the **Find what:** textbox and then click the **FindNext** button. If a match is found it will appear in the program editor with a blue background. The **FindNext** button can be clicked again to find the next instance of the text string and so on ...

To replace a piece of text in the file select **Search | Replace...** from the menu item The various replace options will be presented:

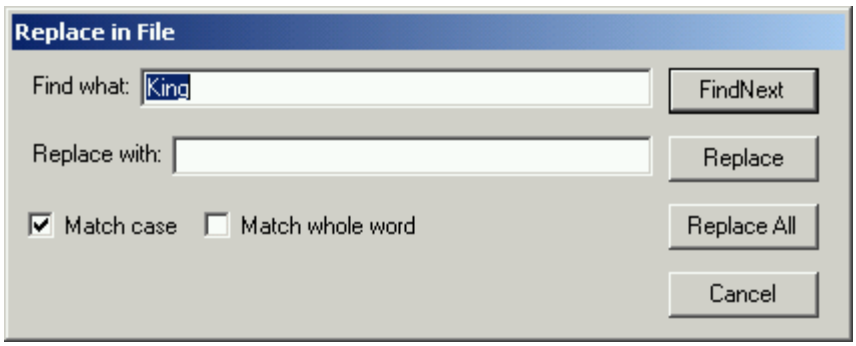
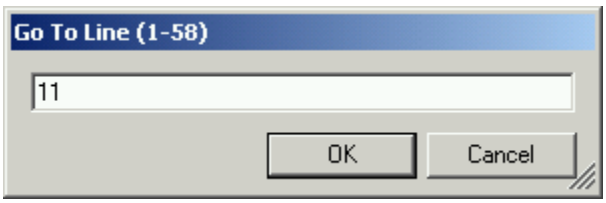


Figure 5.6 The **Replace Options** dialog.

Use the **FindNext** or **Replace** buttons to skip or replace the text instance in the file.

5.3 Line number search

It is also possible to find a particular numbered line in a file, the first line is number 1. To find the line with a given number select **Search | Goto...** in the menu item enter the line number in the dialog box :



5.7 The **Go To Line** dialog box.

The program text will be scrolled and caret will located at the start of line number 11.

6. Compiling a Program

6.1 Setting up the compilation options.

Lets open the project **HelloWorld.zbp** file to compile the project. Before compiling your project or program file you need set the compilation options. To do this select the menu item **Project | Properties...** in the main Zonnon Builder menu:

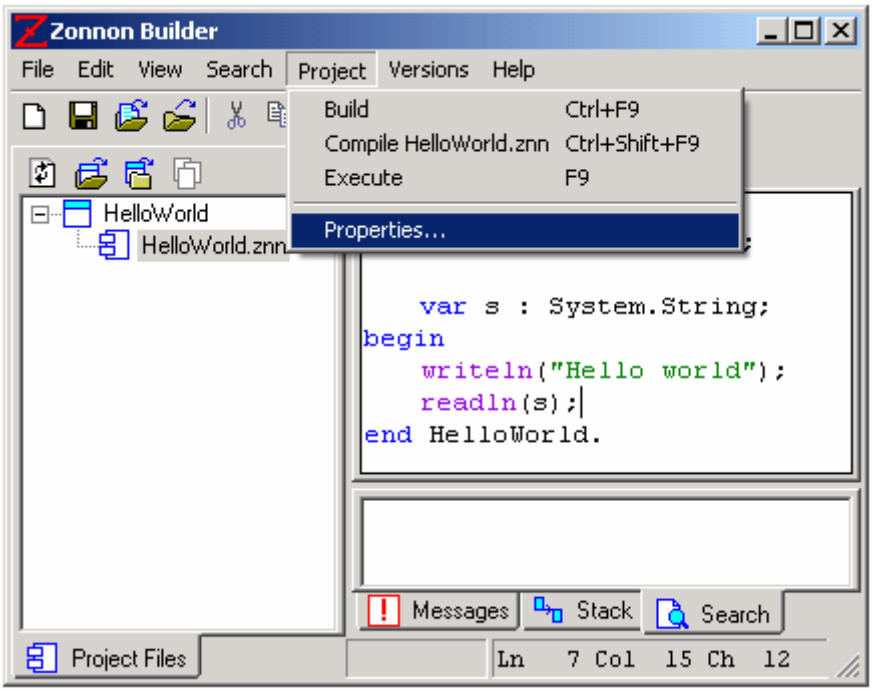


Figure 6.1 The **Project** submenu in main menu.

The project options dialog box will appear :

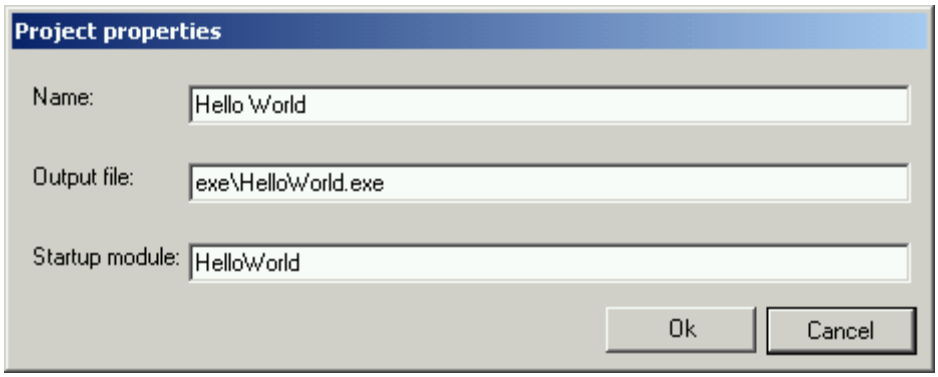


Figure 6.2 The project options dialog box.

Enter the file name for the program to be compiled in the **Output file** text box. In our sample the directory **bin** is located the directory where the project file is located. The program's main module name should be entered in the text box **Startup module**. One of project files must of course contain the Zonnon module named **HelloWorld**.

Now we ready to compile our **HelloWorld** program in the project.

6.2 The project compilation.

To compile the project select **Project | Build** from the main menu, or click on the **Build** icon in the main toolbar.

The results of the compilation of the **HelloWorld** project will now appear in the **Messages** tab of the lower right-hand window :

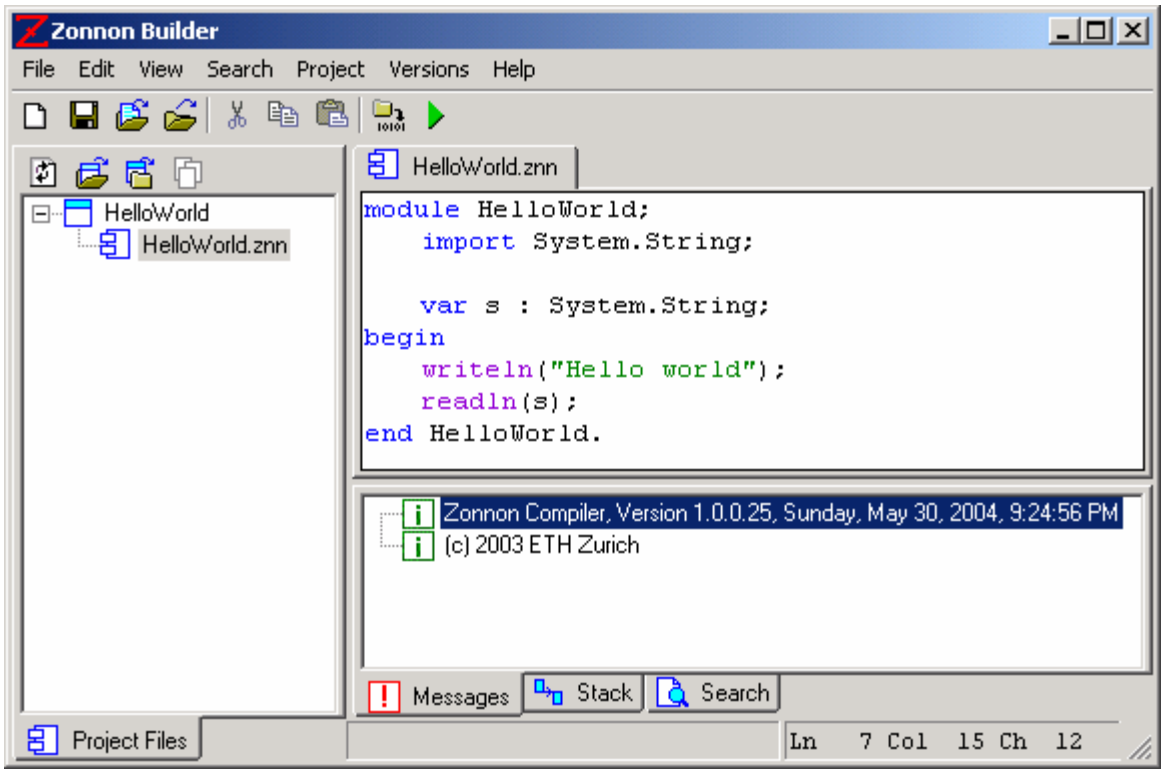


Figure 6.3 A successful **HelloWorld** project compilation.

Lets deliberately introduce an error into the program and see what happens. We will remove the semicolon (;) from the following declaration :

```
VAR s : System.String;
```

Now compile the project again and then look at the compiler diagnostics in the **Messages** tab :

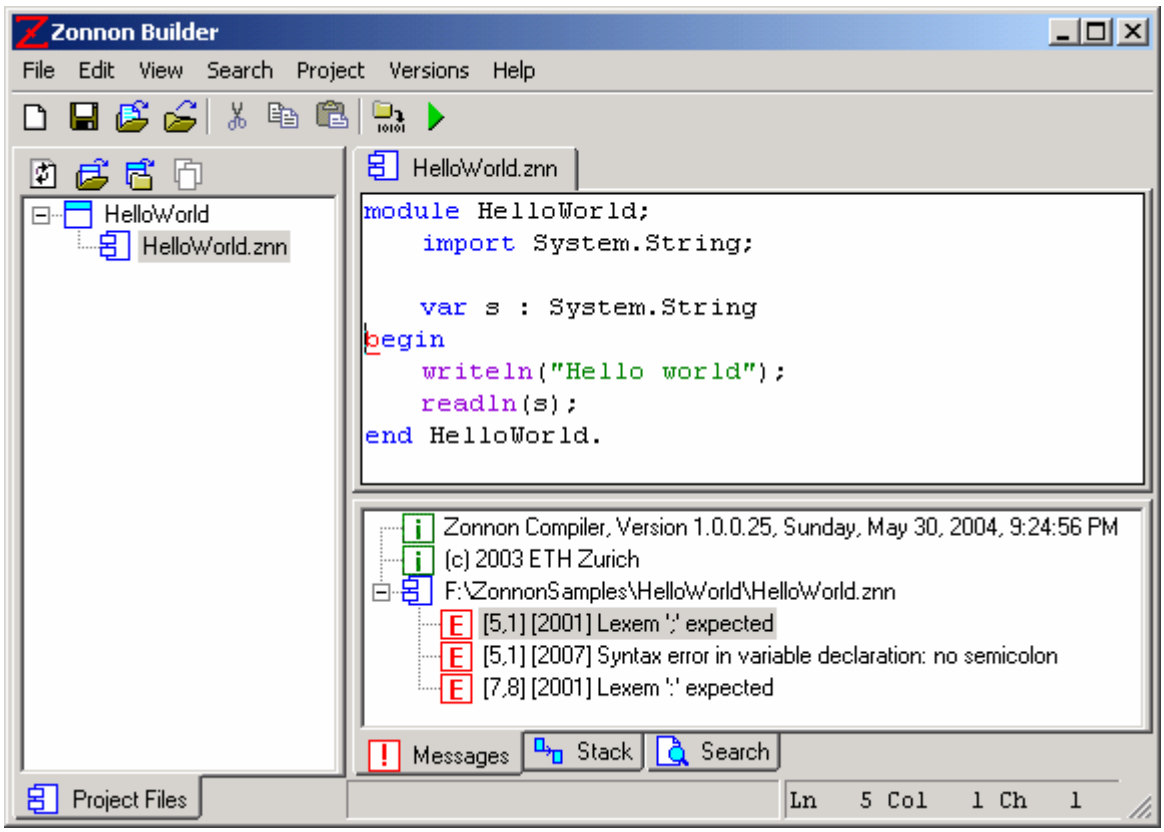


Figure 6.4 The **HelloWorld** project compilation showing errors.

Now double click on first error diagnostic in the **Messages** tab :

[6,1] [2001] Lexem ';' expected

The file **HelloWorld.znn** opens with the text scrolled to the error location **[6,1]** which will be highlighted in red :

```
BEGIN
```

6.3 The program file compilation.

It is also possible to compile an individual file using the editor tab's popup menu **Compile** :

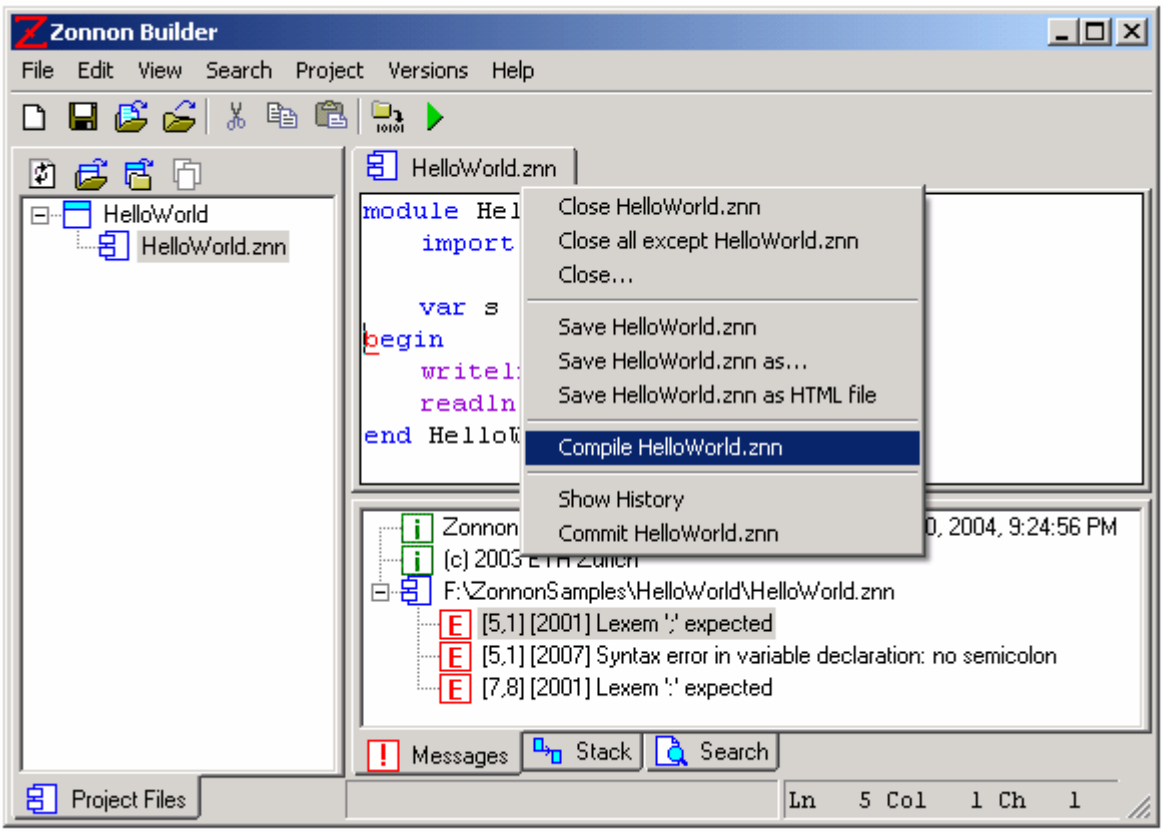


Figure 6.5 **HelloWorld.znn** file compilation using the editor tab popup menu.

Another possibility is to use a popup menu of the relevant project tree file node:

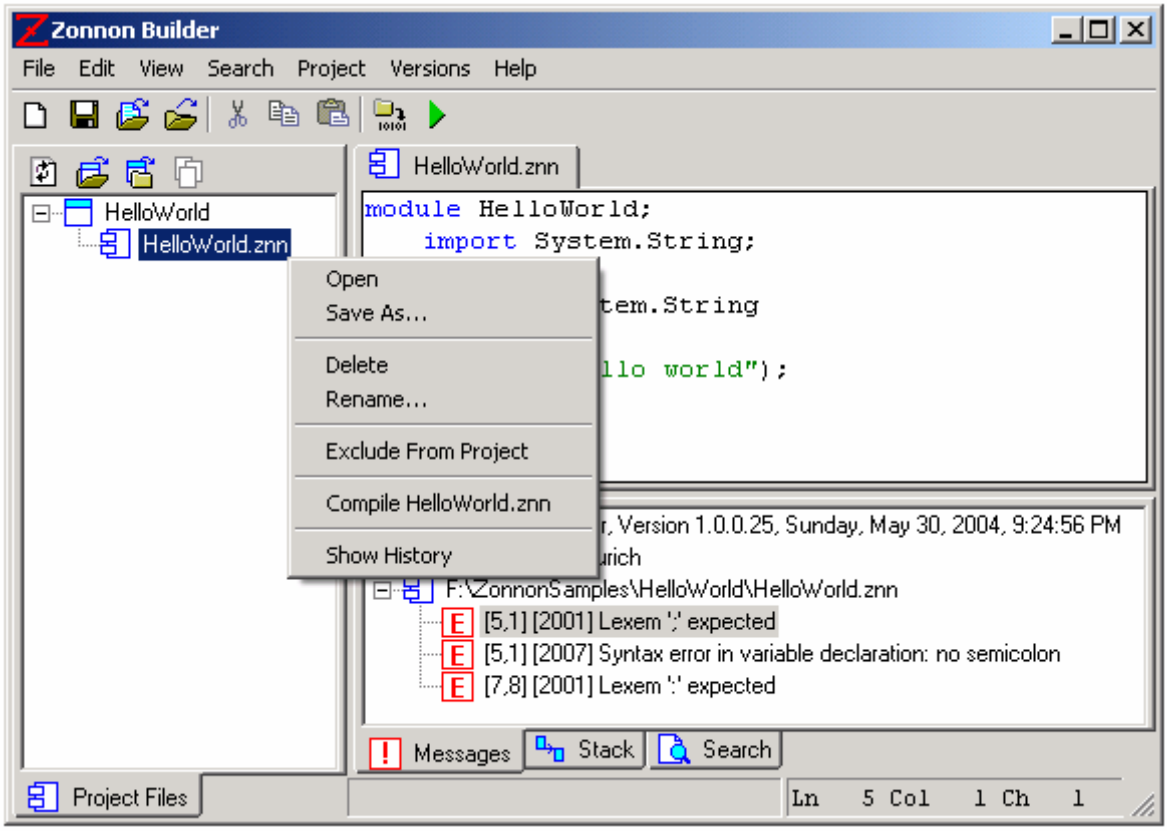


Figure 6.6 **HelloWorld.znn** file compilation using a project tree file node popup menu.

Now the program is ready to run, so lets do that next.

7. Executing a Program

7.1 Successful program execution

Once a program has been successfully compiled it can be executed. So lets execute the **HelloWorld** program by clicking on the menu item **Project | Execute,** or click on the **Run** icon in the main toolbar, or select the **Execute** menu item of the project tree root node:

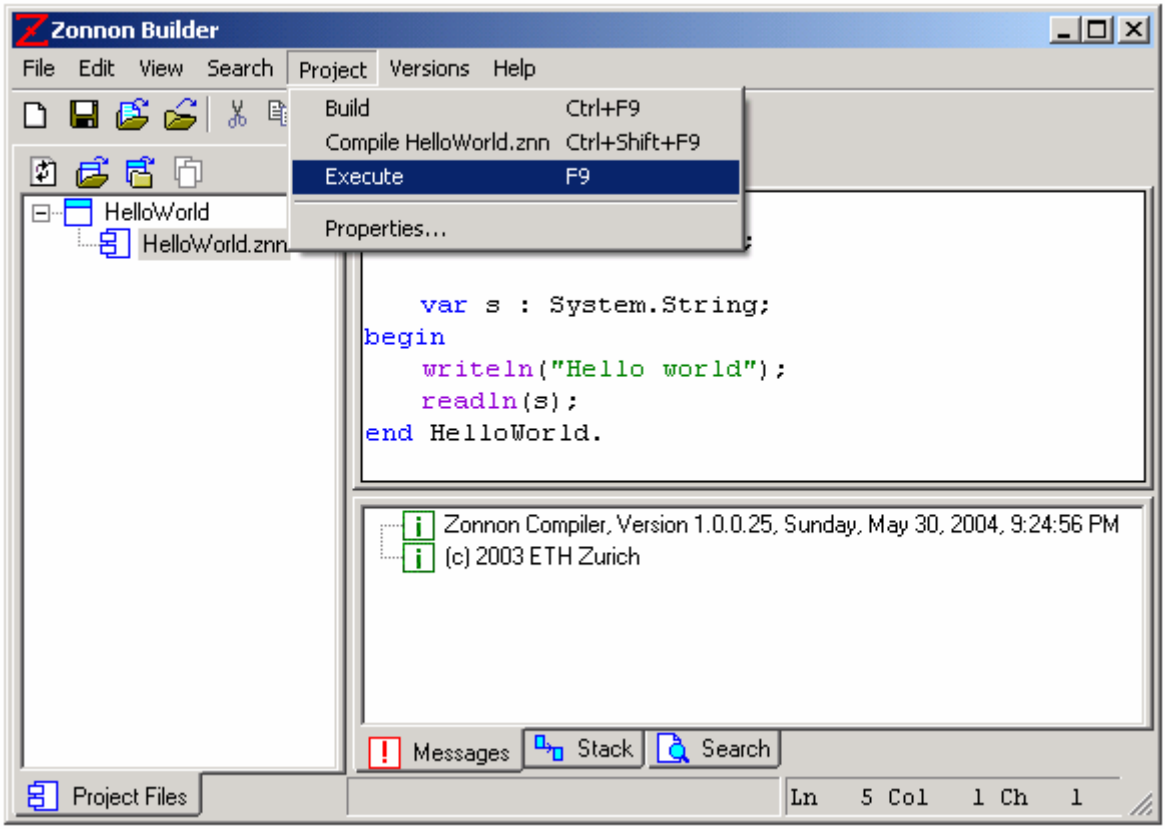


Figure 7.1 The **Execute** menu item in the project tree root node.

When the program starts to run the **HelloWorld** console window appears :

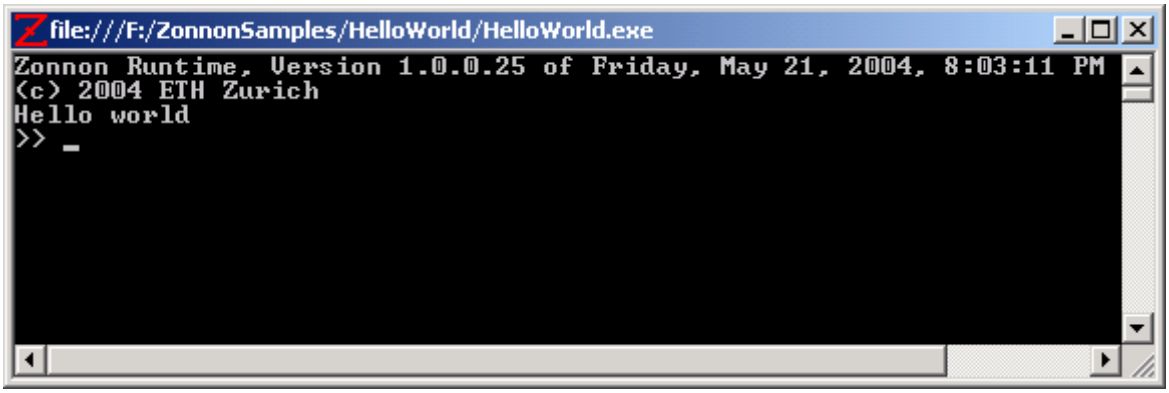


Figure 7.2 The **HelloWorld program's** Console Window.

The **HelloWorld** program will be executed and it will wait for any text to be entered before running to completion. The execution results will appear in the **Messages** tab :

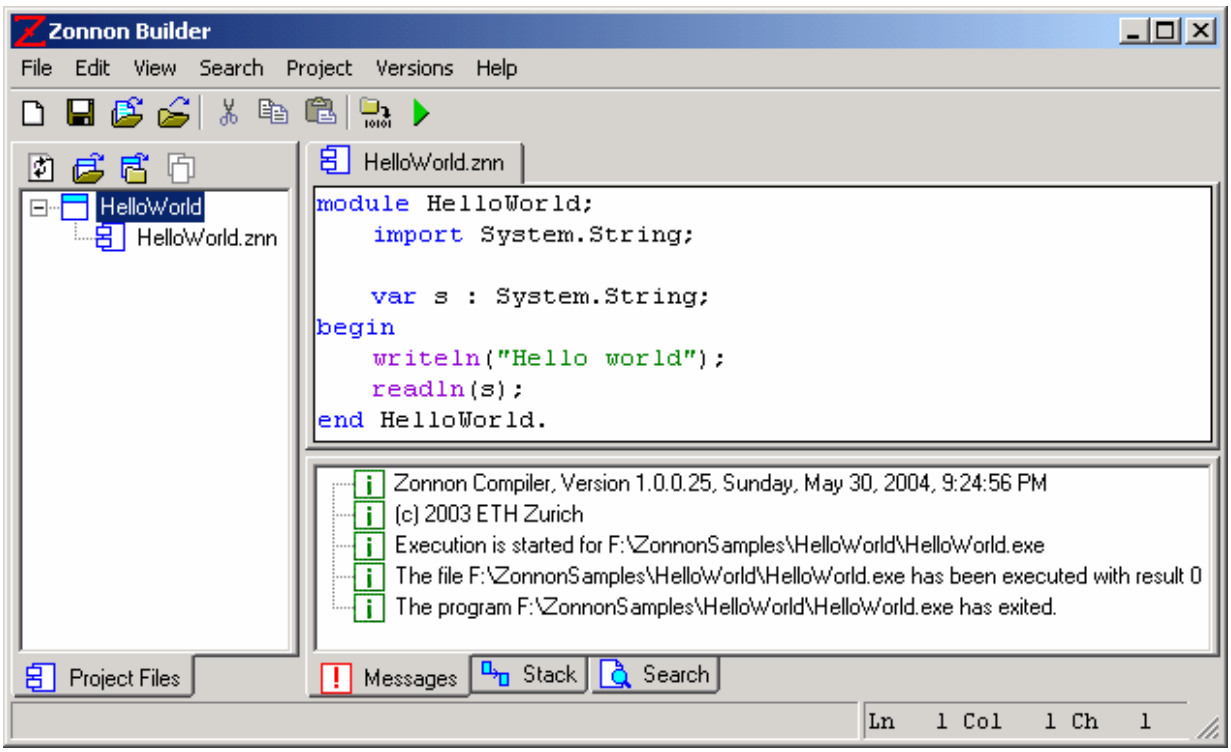


Figure 7.3 The **HelloWorld** program execution result.

7.2 Post-mortem program stack

Whilst the program is running the Zonnon Builder keeps a trace record of the program execution stack. The project **StackView.zbp** project is designed to reveal the contents of the (post-mortem) stack when a program has been terminated or run to completion. After program compilation and execution the stack information appears in the **Stack** tab of the lower right-hand window :

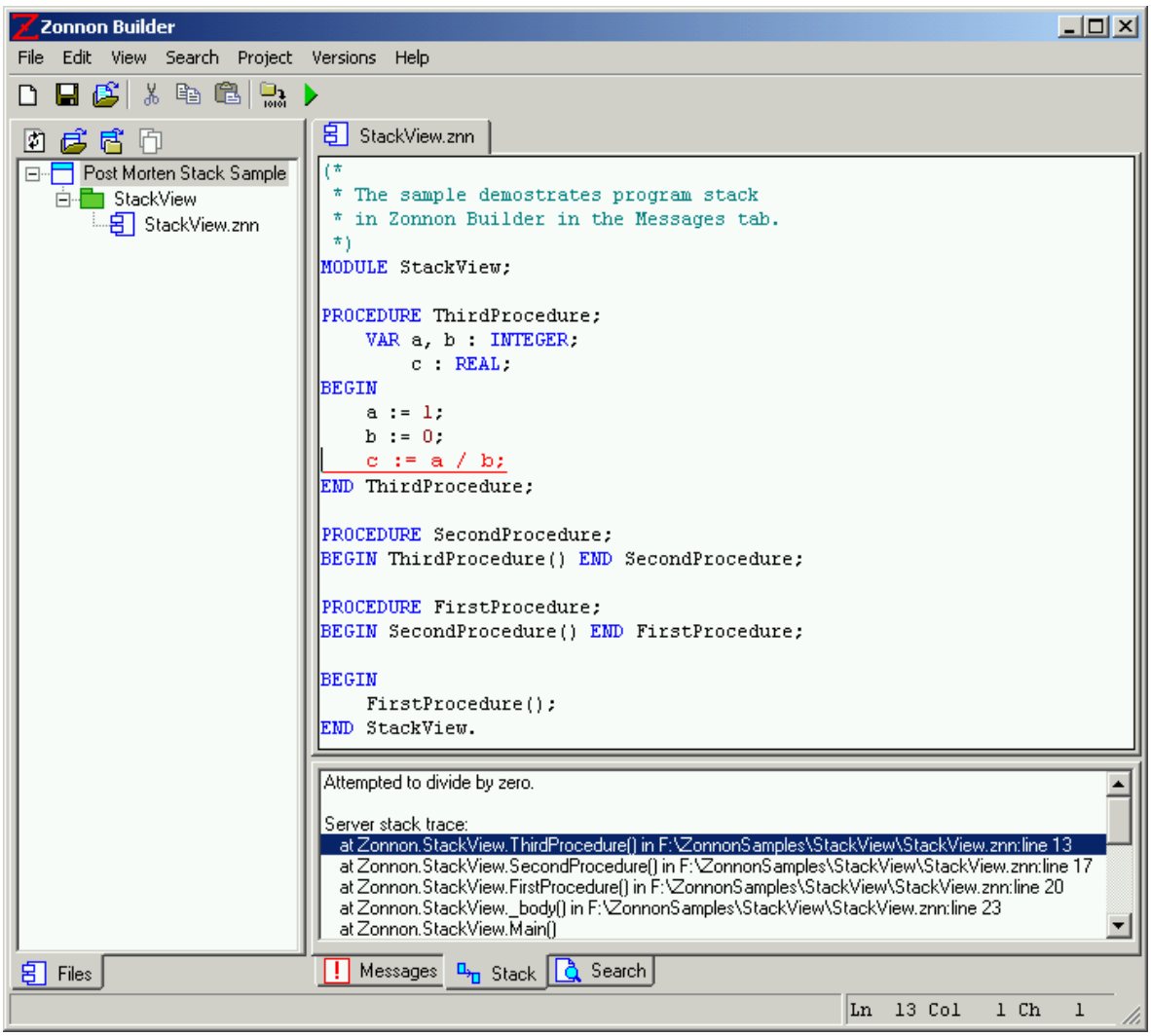


Figure 7.4 The **StackView** post-mortem stack in the **Stack** tab.

8 Setting Up a Project

8.1 Creating a Project

The Zonnon Builder uses the notion of a **project** to organise a set of related Zonnon **program** files.

When Zonnon Builder starts up you will see the initial empty window :

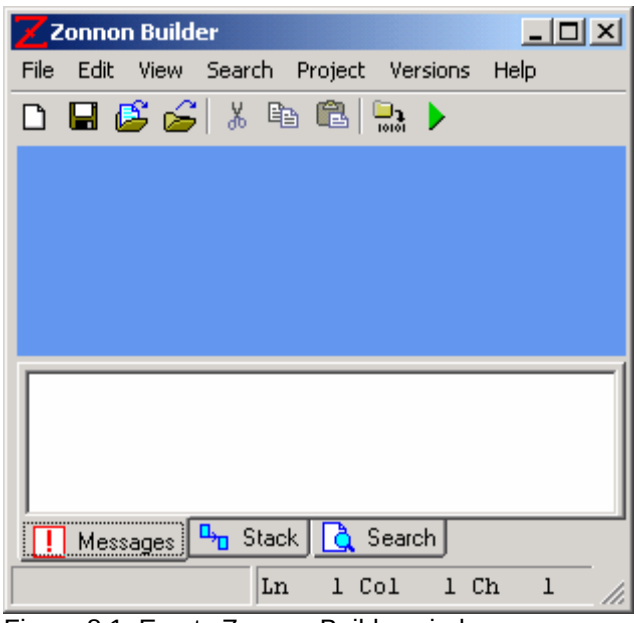


Figure 8.1 Empty Zonnon Builder window.

To create new project turn on project panel by menu item **View | Project Files**.

The first step is to create a **project** to store and organise the **program** files.

Start by clicking the right mouse button on the project icon in project files tree.

The following popup menu will appear :

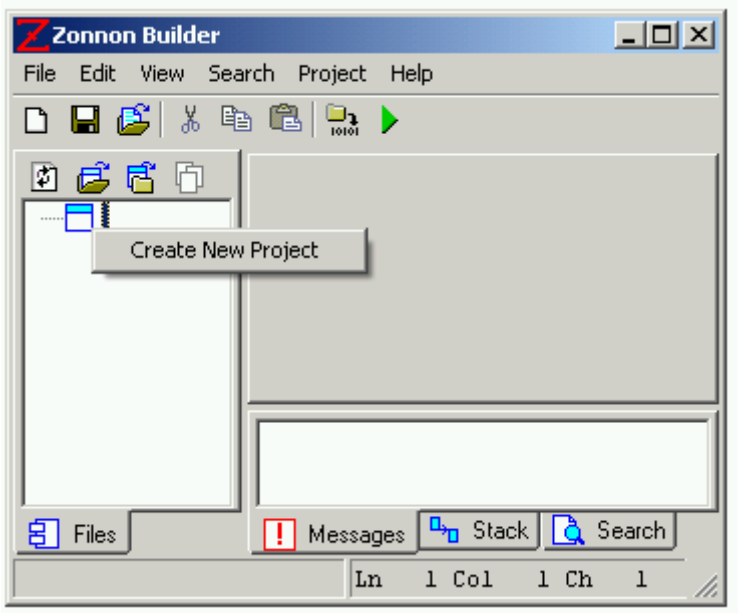


Figure 8.2 Popup menu for an empty project.

Another way to do this would be to select **File | New Project** from the main menu item as shown below :

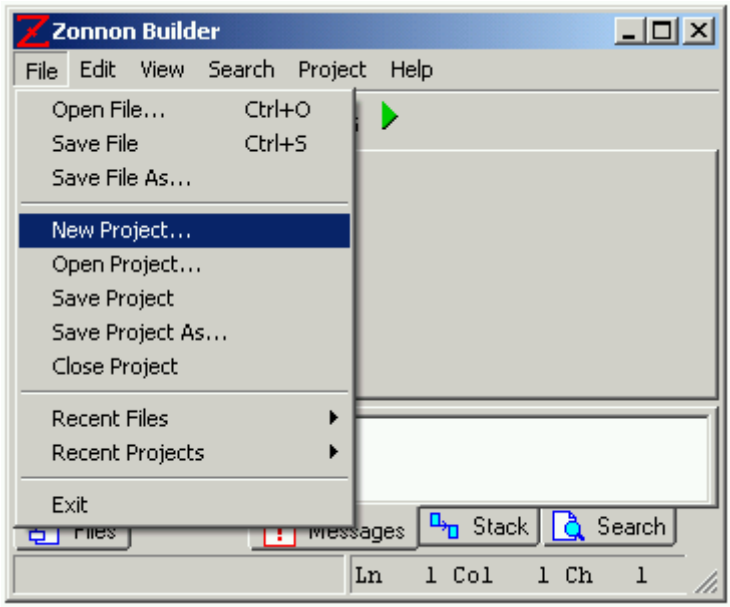


Figure 8.3 **New Project...** main menu item.

Now select a directory name for the new project and then enter the project filename :

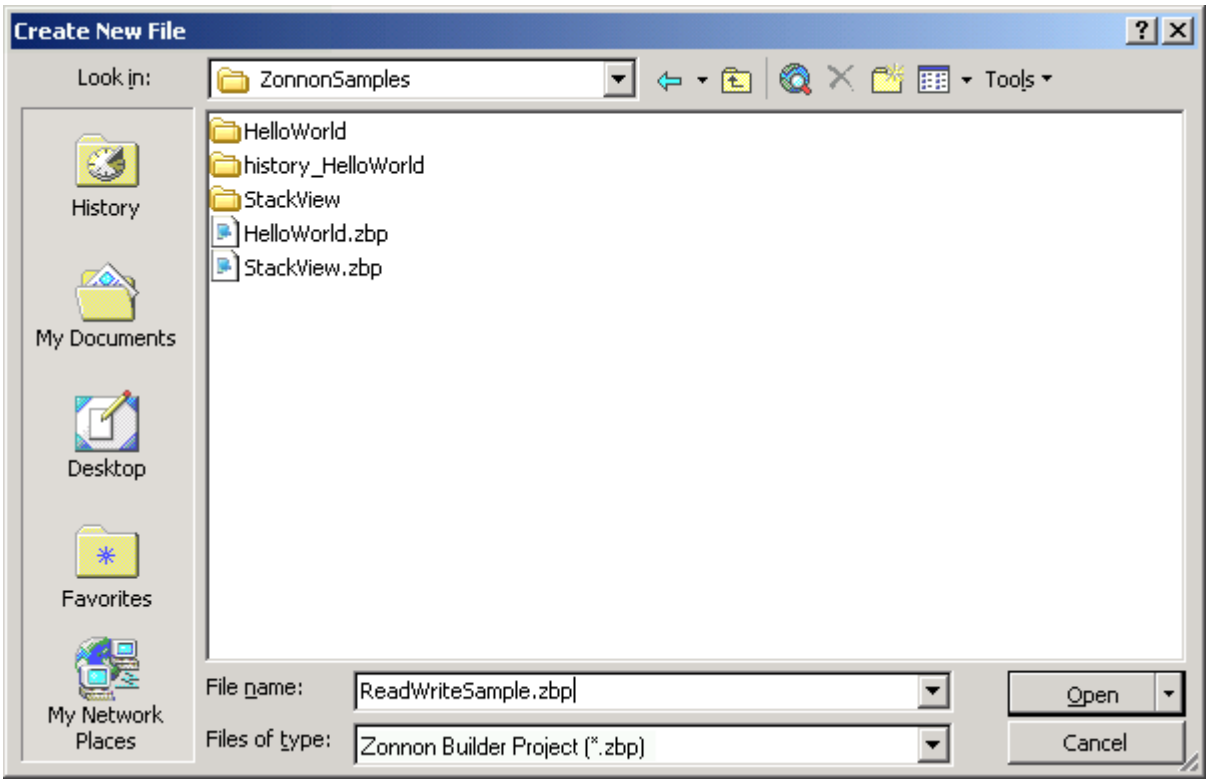


Figure 8.4 Creating the Project File

When the new project file name **ReadWriteSample.zbp** has been entered the project is automatically given the default name **ReadWriteSample** :

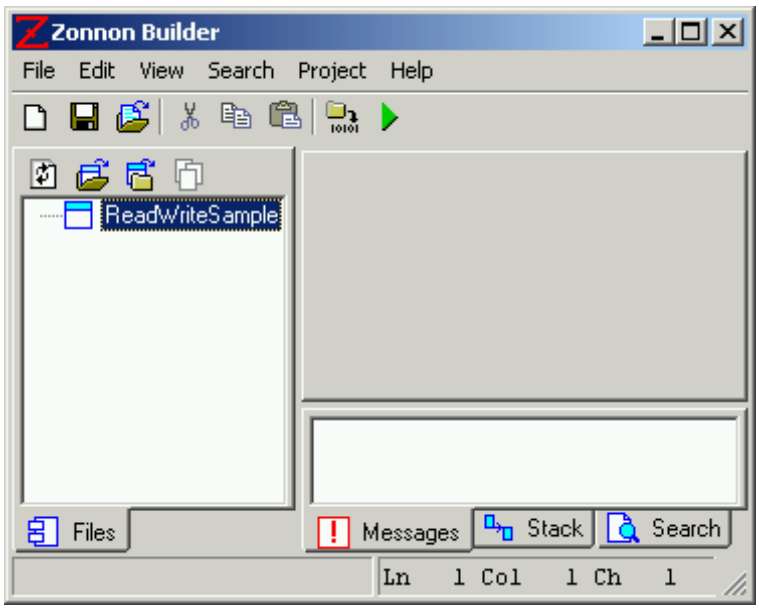


Figure 8.5 An empty project with the name **ReadWriteSample**.

8.2 Setting up the Project File Structure

Lets create some folders and files for the new project. To do this review the directories and files that already exist by clicking the left mouse button on **the ShowAll** icon in the files toolbar. It is located in the left hand window pane just above the project name.

You can now see all the directories and files in the project file directory and its nested directories. The directories and files that are not included in the project are coloured grey :

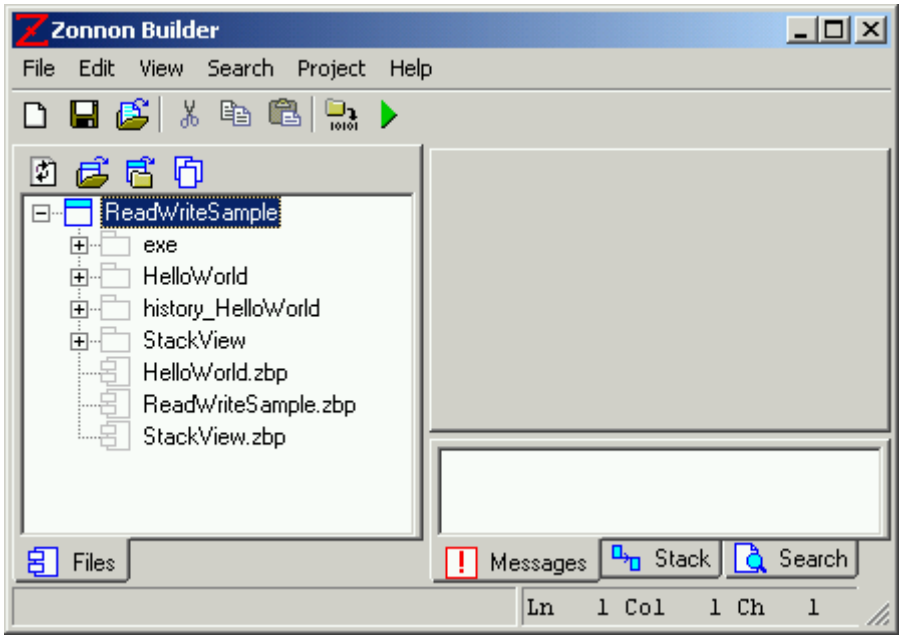


Figure 8.6 All directories and files.

Lets create our own directory for the **ReadWriteSample** project. Click the right mouse button on the project icon and select the menu item **Add New Folder**.

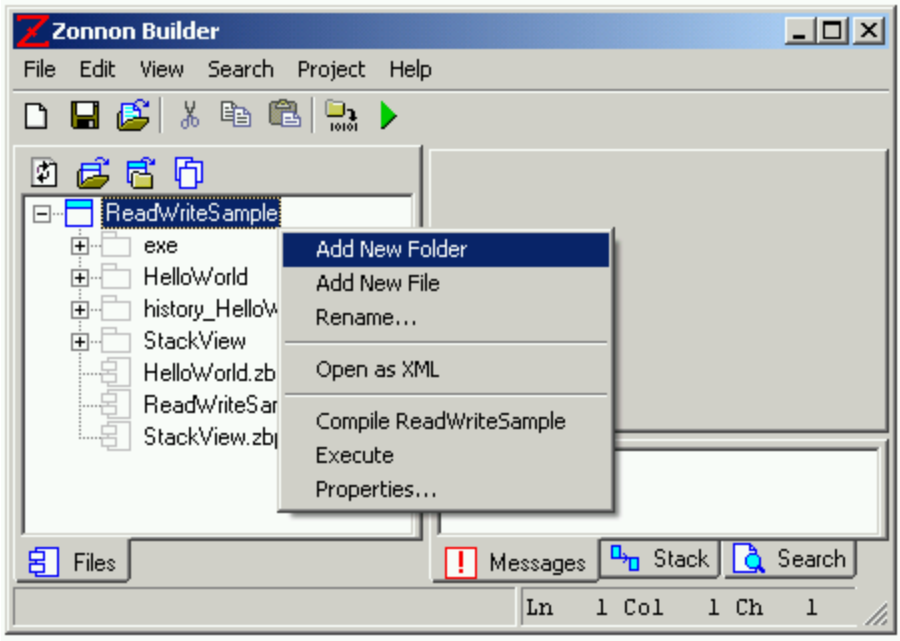


Figure 8.7 Project icon popup menu.

Enter the new folder name **ReadWrite** in the dialog box. A new green folder icon will appear in the project tree, as shown below :

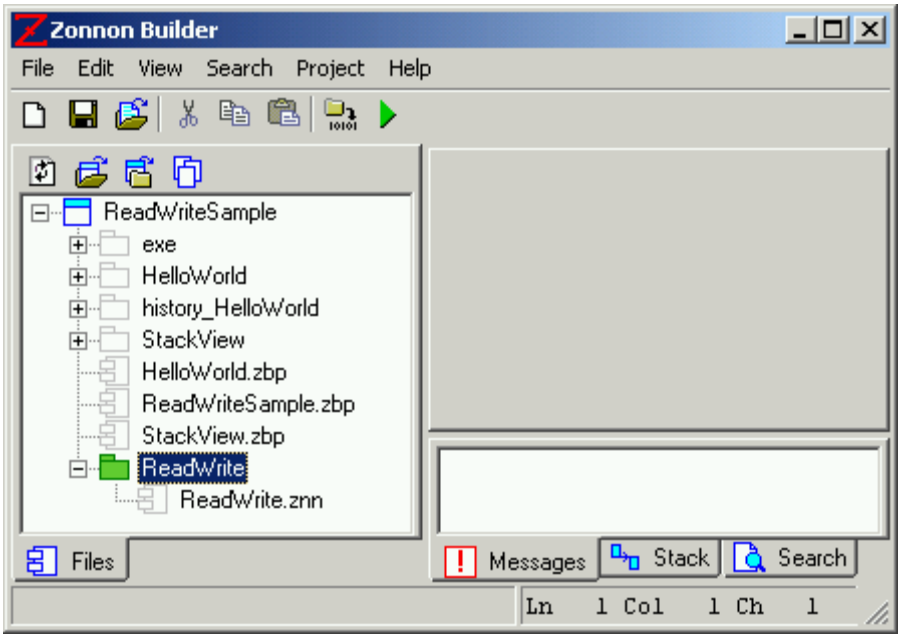


Figure 8.8 New file **ReadWrite** in the project tree.

Now press the right mouse button on this new folder icon and select the menu item **Add New File**. Enter the new folder name **ReadWrite.znn** in dialog box. The **.znn** ending to the file name indicates that the file contains Zonnon program source text. Now you will see the popup menu shown below, this can be used to include any file or exclude any file from the project :

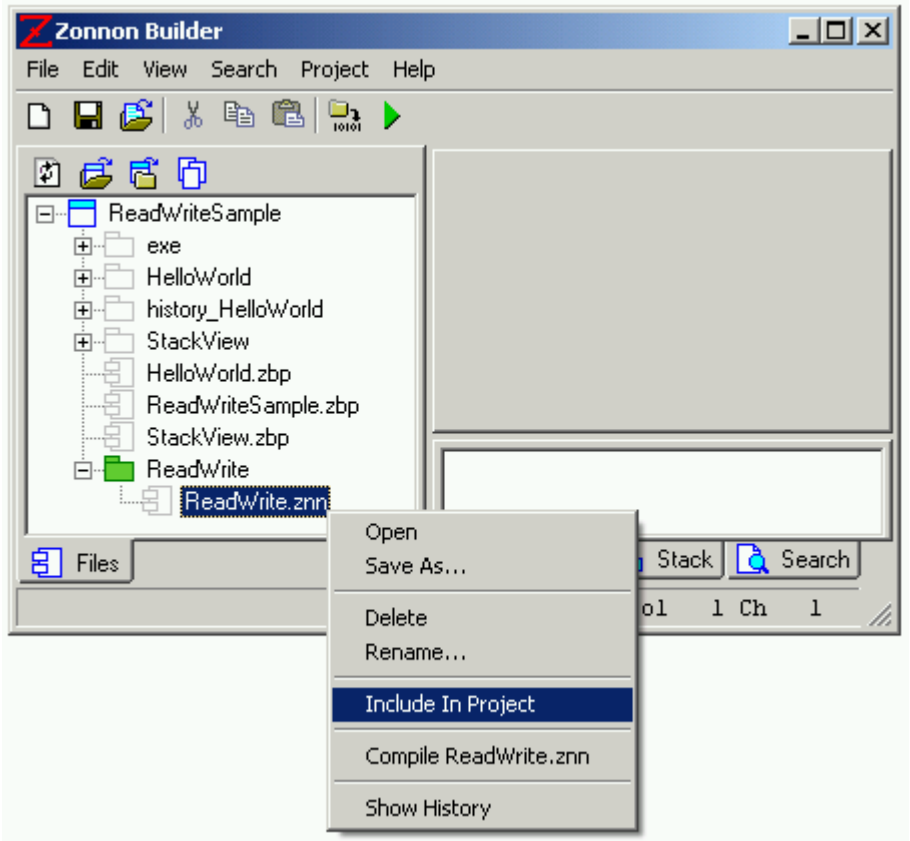


Figure 8.9 File popup menu.

The file will be included in the project and it will have the source file symbol as an icon .

Now click on the **Show Included Only** icon in project tree toolbar.

Here is the project tree for the project :

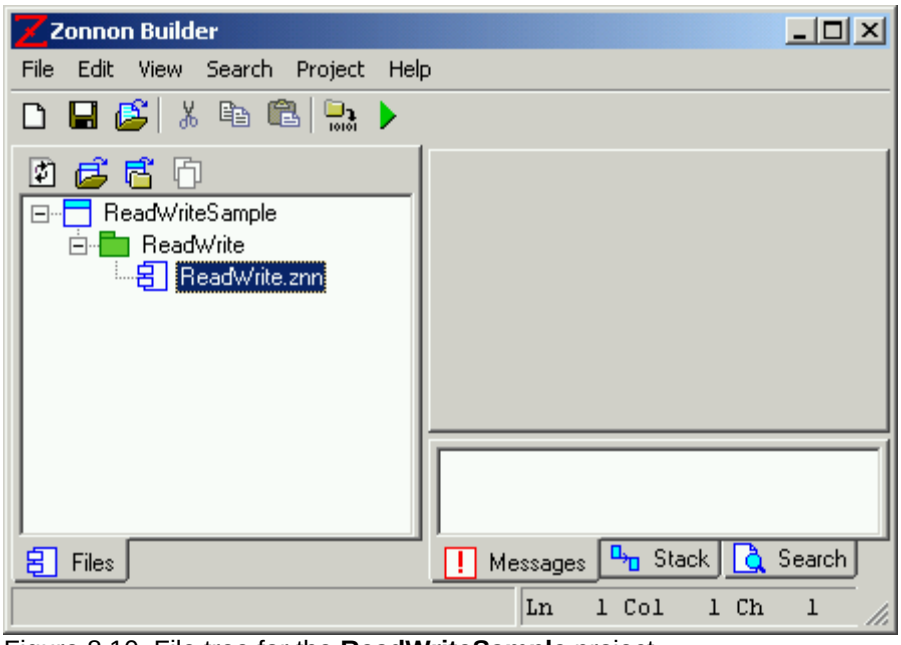


Figure 8.10 File tree for the **ReadWriteSample** project.

Double clicking on the **ReadWrite.znn** file in the project tree will now open it in the Zonnon syntax oriented editor.

8.3 Opening and Re-opening Projects.

Lets open the project **ZonnonChess** as an example. First of all press the **Open Project** icon in project tree toolbar or select the main menu item **File | Open Project...**.

Then use the standard .NET dialog box to select the Zonnon project file, this reveals the structure of the **ZonnonChess** project :

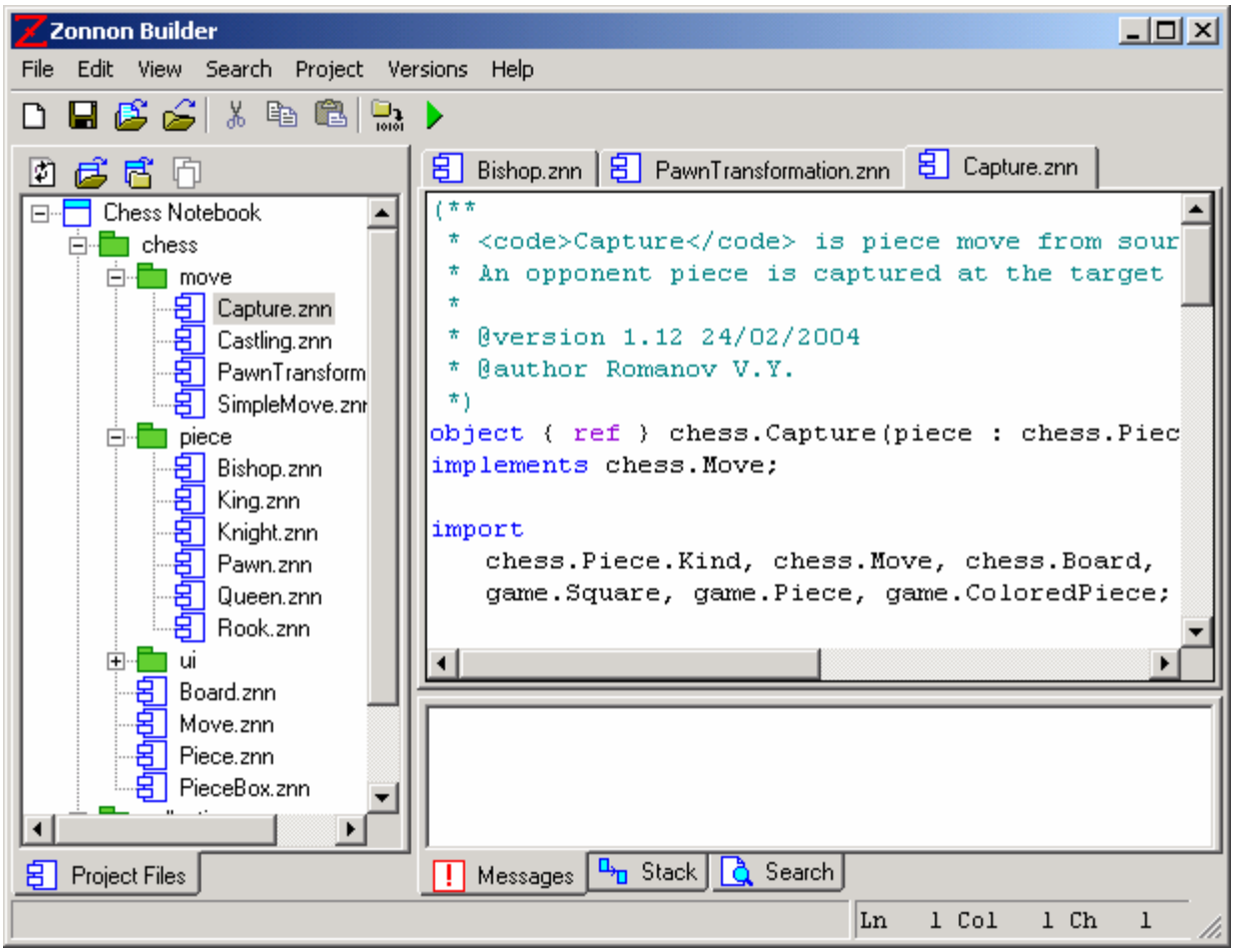


Figure 8.11 **ZonnonChess** Project structure.

Zonnon Builder records the history of project openings. Any Zonnon project can simply be re-opened by clicking on the **Reopen Project** icon in the project tree toolbar or by selecting the main menu item **File | Recent Projects**.

A list of recently used projects will appear :

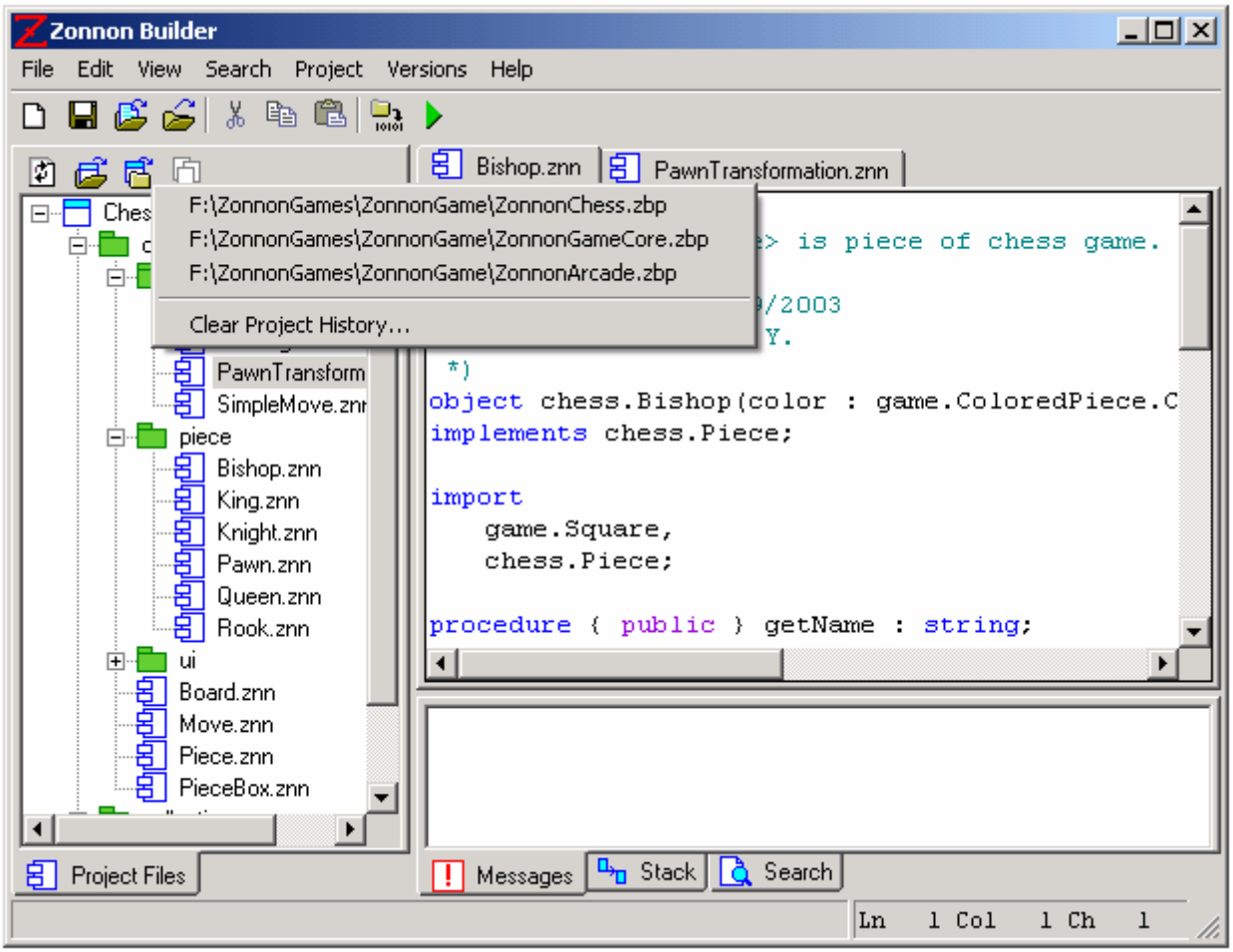


Figure 8.12 The **Recent Projects** list.

If the re-open list becomes too long to be manageable then some elements can be removed from the list. To do this select the item **Clear Project History...** in the list and then tick the check boxes of the projects are to be removed from the list by clicking on them :

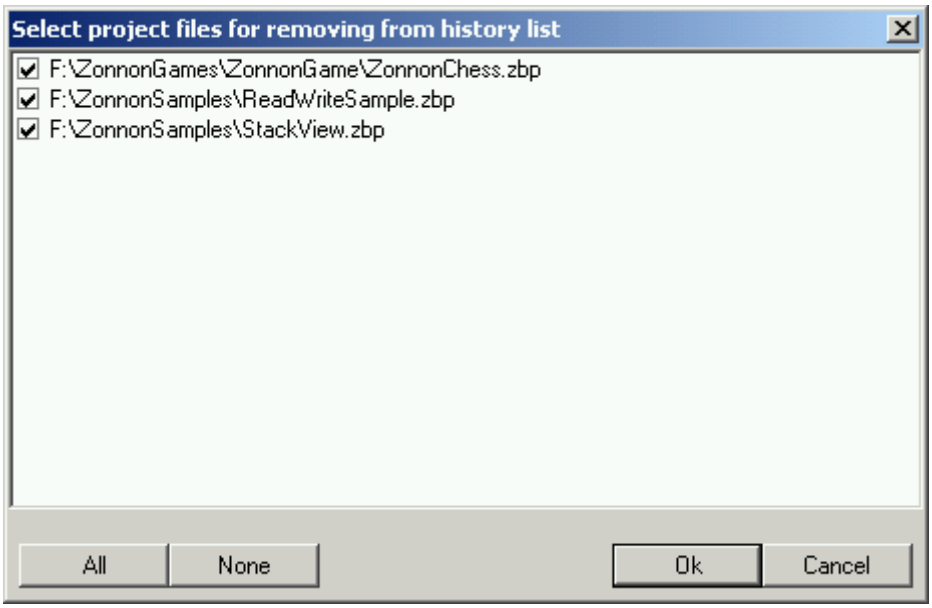


Figure 8.13 **Clear projects history** dialog box.

The next step is to create and edit programs in Zonnon syntax oriented editor.

9. Versioning. Introduction

9.1 Versioning terms and concepts

The Zonnon Builder keeps a history of your project files. Every time a file is changed and saved a copy of it is also saved and these file copies are called file *revisions*. It is possible to restore back to any file revision state in the file's history.

Once a file has reached some notional of 'completion' it can be 'committed' as a *file version*. For example you might define a new file version if the file compiles without errors, or perhaps when some algorithm has been successfully implemented and tested. The file version has a unique *version number*, for example, the number 1.3. It is also possible to attach long text as part of the file version. The *version number* is used to denote the version difference. The long text will called *version comment* and it contains a description which relates to the file changes. The file *versions* make up one of the dimensions of versioning.

Some new feature of your program may be implemented by editing *several* files in the *project*. After it has been debugged a new version number can be defined for a *project version,* a short label and a long comment can also be added. During the development process each file may be changed many times and have one or more *file versions*. If any file has been changed but does not have *file version* number then it will be given the next *file version* number as a default. The project version number becomes the .NET *assembly version* number when the project is compiled. This makes it feasible to restore the project configuration for each .NET assembly that has been deployed to the program's users.

The current Zonnon Builder has simple implementation of these versioning concepts. All revisions and versions are located on your computer. In future other versioning implementations may use popular distribution version control systems as the basis for the versioning implementation, for example PVCS, QVCS, CVS and SourceSafe.

9.2 File revisions and versions

To view a history file select the **Show History** menu item in the file's popup menu in the project tree :

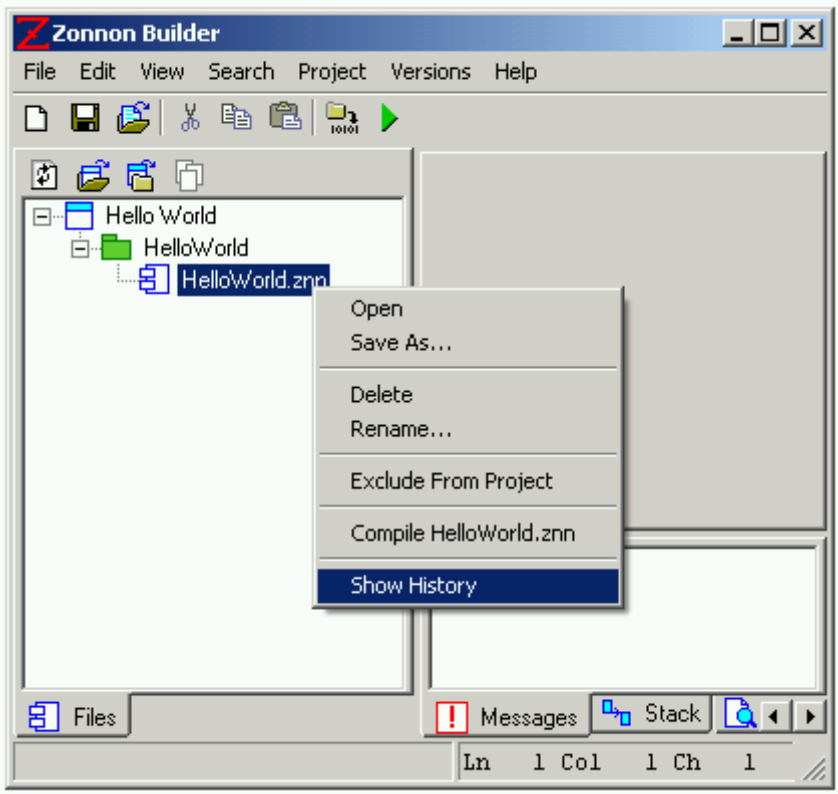


Figure 9.2.1 The **Show History** item in the project tree's menu.

The file history window appears in the Zonnon Builder's centre window, it has three tab windows: **Contents**, **Diff** and **Info**.

In the **Contents** tab window the top panel contains the file history list, where the top list item presents current file. Current file item has a **File icon** . The next items are the versions and revisions of the file, each file version has a **File Version icon** and each file revision has a **File Revision icon** .

Every file history list item has a data and a time attribute.

If you select a file, version or revision in the history list the relevant program text will be presented in the lower panel of **Contents** tab window. The status panel will show the date and time attributes of the item. The program text and attributes for the **Version 1.2** of file **HelloWord.znn** are shown below :

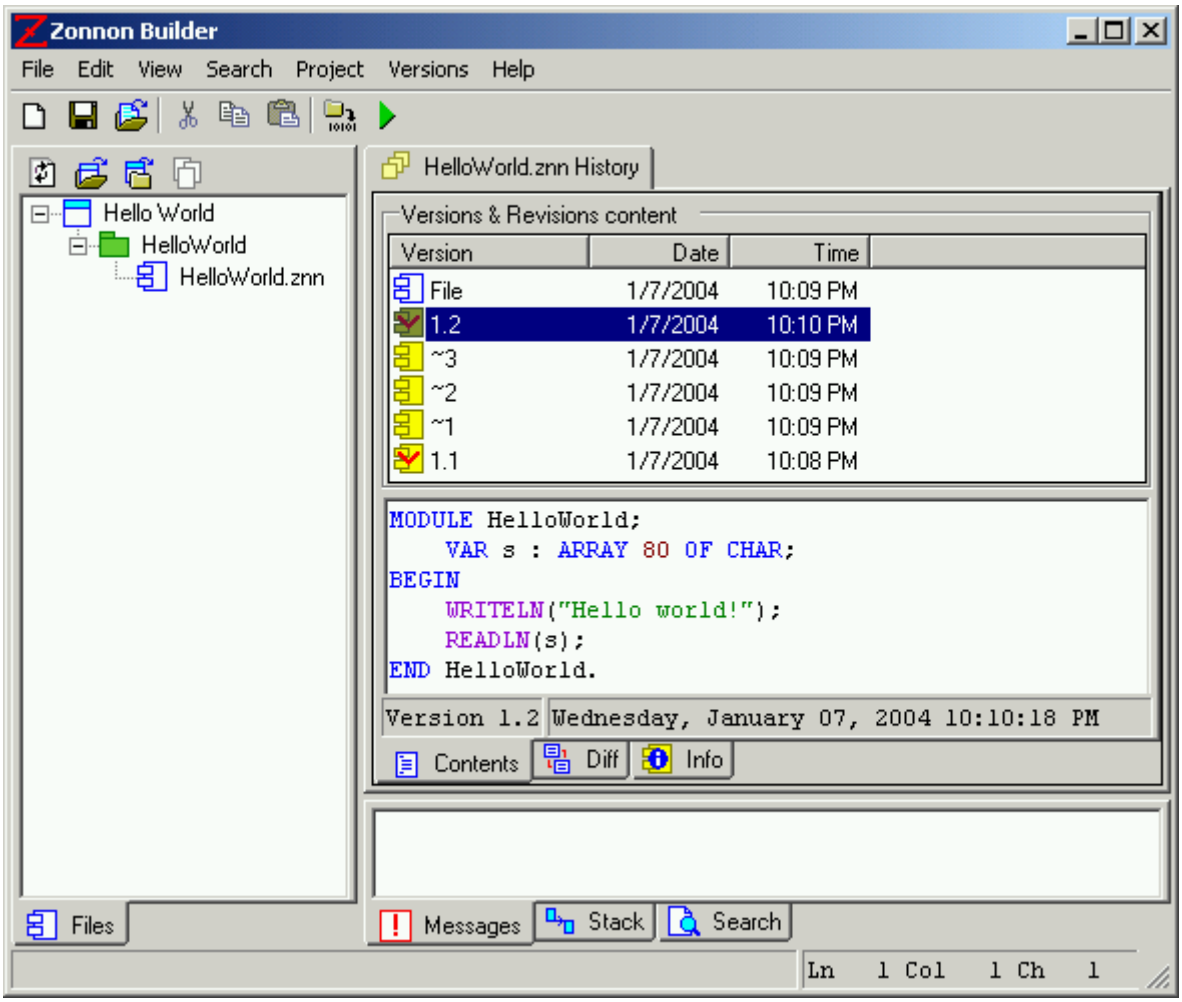


Figure 9.2.2 The program text for the **Version 1.2** of file **HelloWord.znn**.

Similarly the program text and attributes for the **Version 1.1** of file **HelloWord.znn** are shown below :

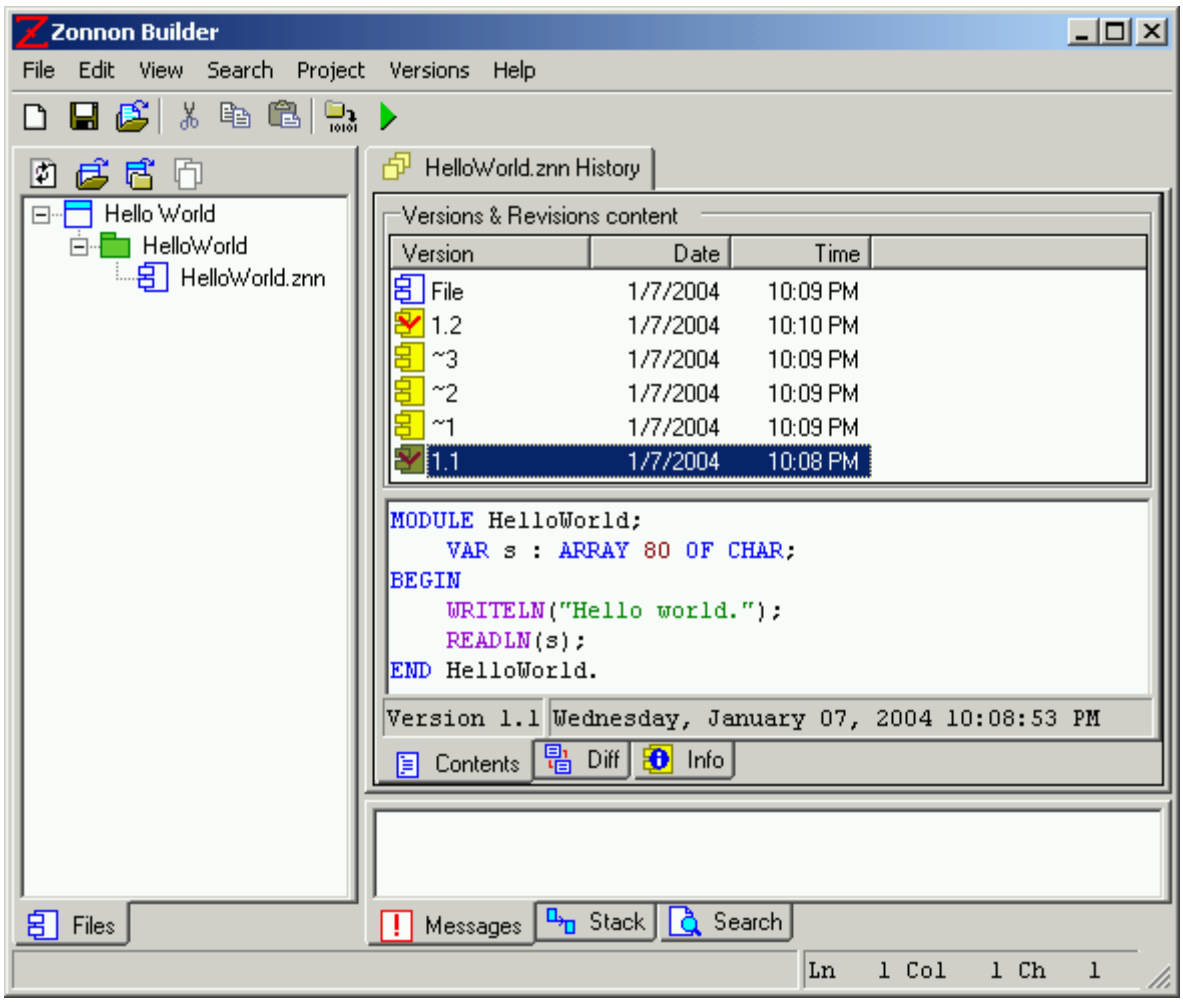


Figure 9.2.3 The program text for the **Version 1.1** of file **HelloWord.znn**:.

Now lets look at the **Diff** tab window. The top panel contains two file history lists which make it easy to compare the history of the program texts in the lists. The differences between the file's versions and revisions is shown in the lower panel. The program lines which are common to both files have a white background. Program lines that differ between the files have different background colours. A red background colour highlights items from the first history list and a green background colour highlights items from second history list. The status line shows the number of compared items and the difference count. This example illustrates the difference between **Version 1.1** and **Revision 1** of the **HelloWord.znn** file.

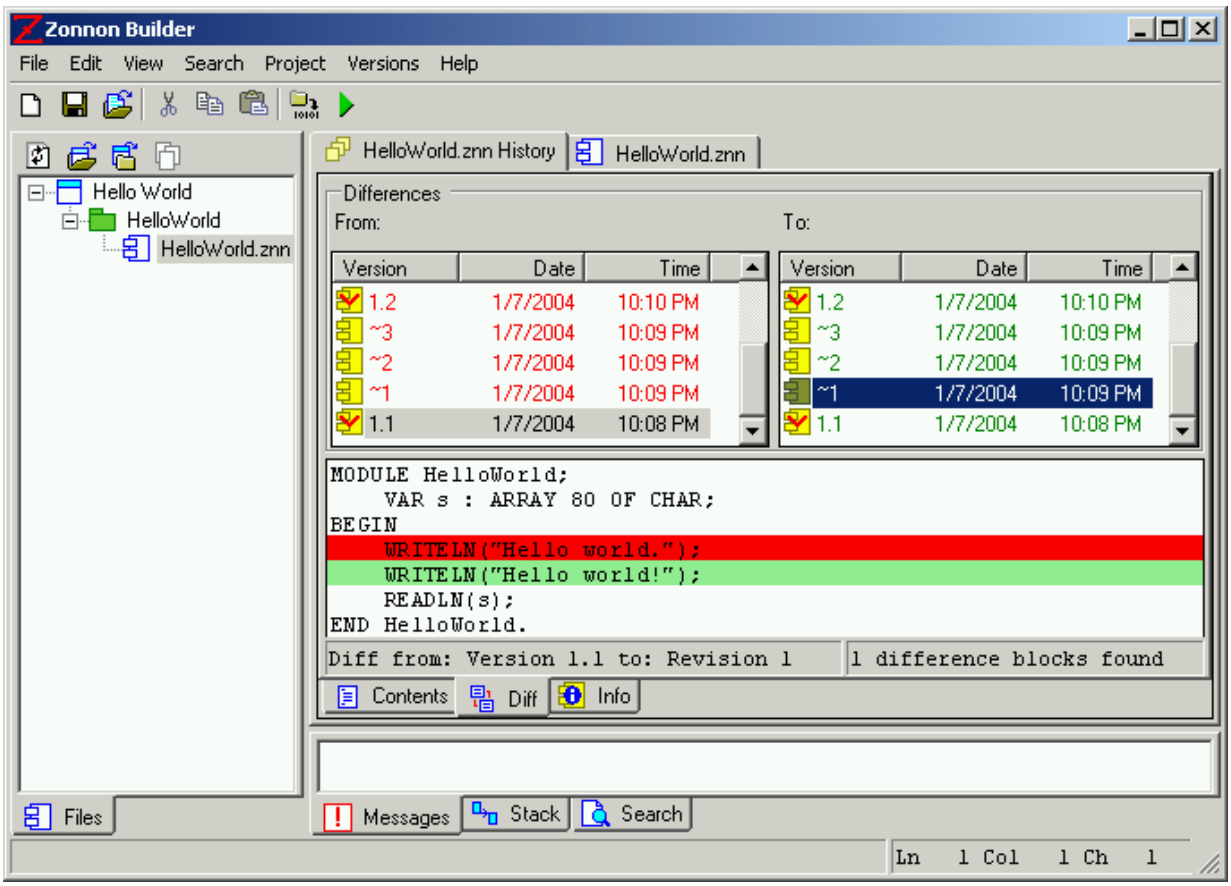


Figure 9.2.4 The differences between **Version 1.1** and **Revision 1** of the **HelloWord.znn** file.

Lets look at the **Info** tab window which shows information about the file versions. The top panel contains the history list including the file versions, whilst the middle panel contains the label. The label gives short explanaition for the file version. The bottom panel contains the comment from the file version. The following Figure shows the version label and comment for **Version 1.2** of the **HelloWord.znn** file :

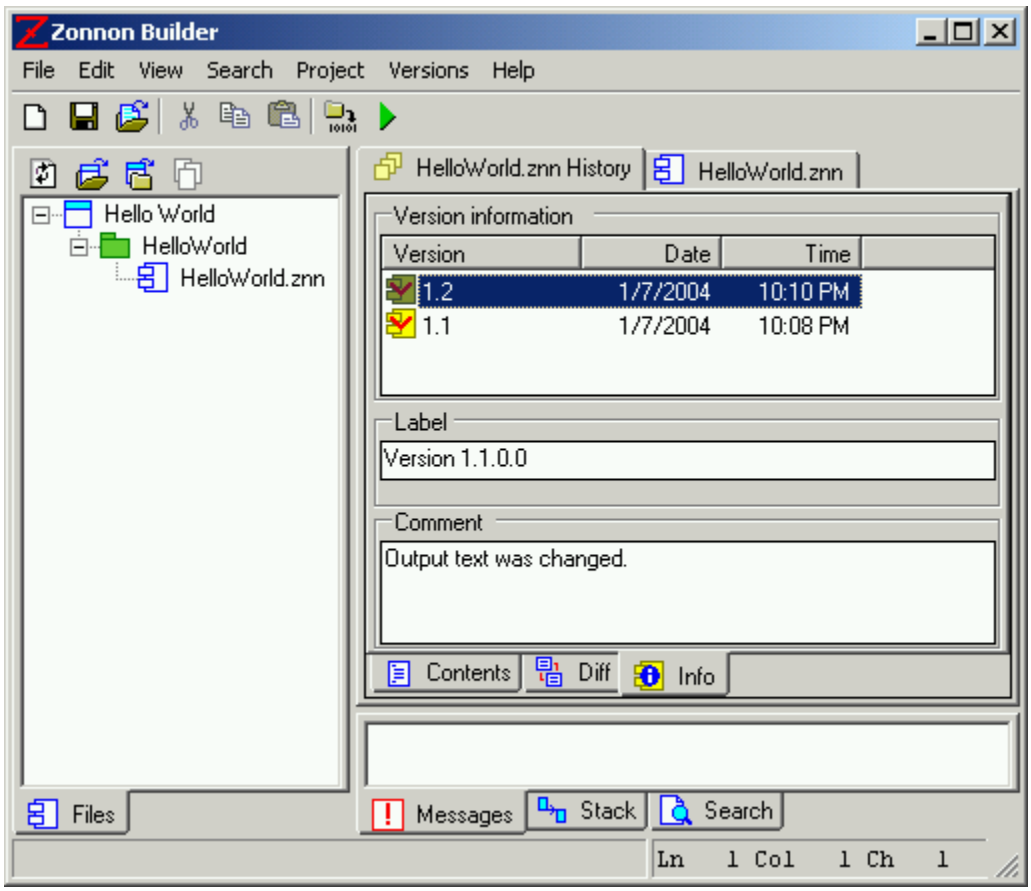


Figure 9.5 The information for **Version 1.2** of **HelloWord.znn** file.

Whenever a file is saved the next *file revision* will be created for the file. To refresh history lists just click the **Refresh** menu item of the **History Window** popup menu:

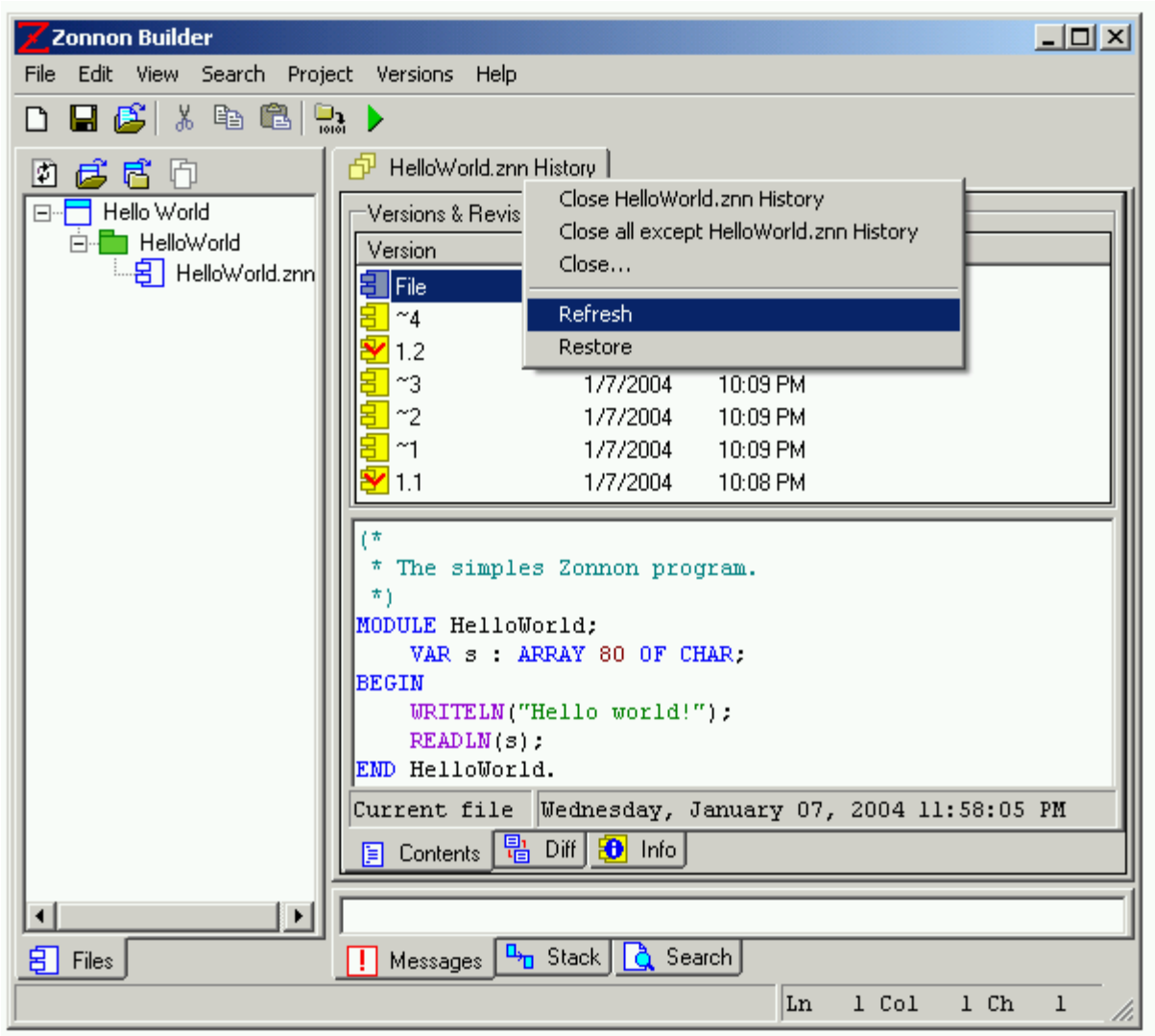


Figure 9.2.6 The **Refresh** menu item of the **History Window** popup menu:

The new file revisions and versions will appear in the history lists.

When the file content is in an appropriate state (i.e. compiled or debugged) the file content can be committed as the next file version. To do this select the **Versions | Commit File** menu item in the main menu :

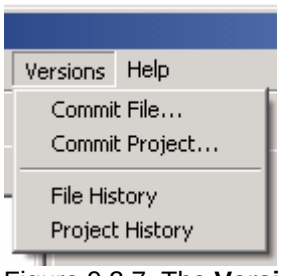


Figure 9.2.7 The **Versions | Commit File** menu item on the main menu

It is also possible to select the **Commit HellowWorld.znn** in the program editor popup menu :

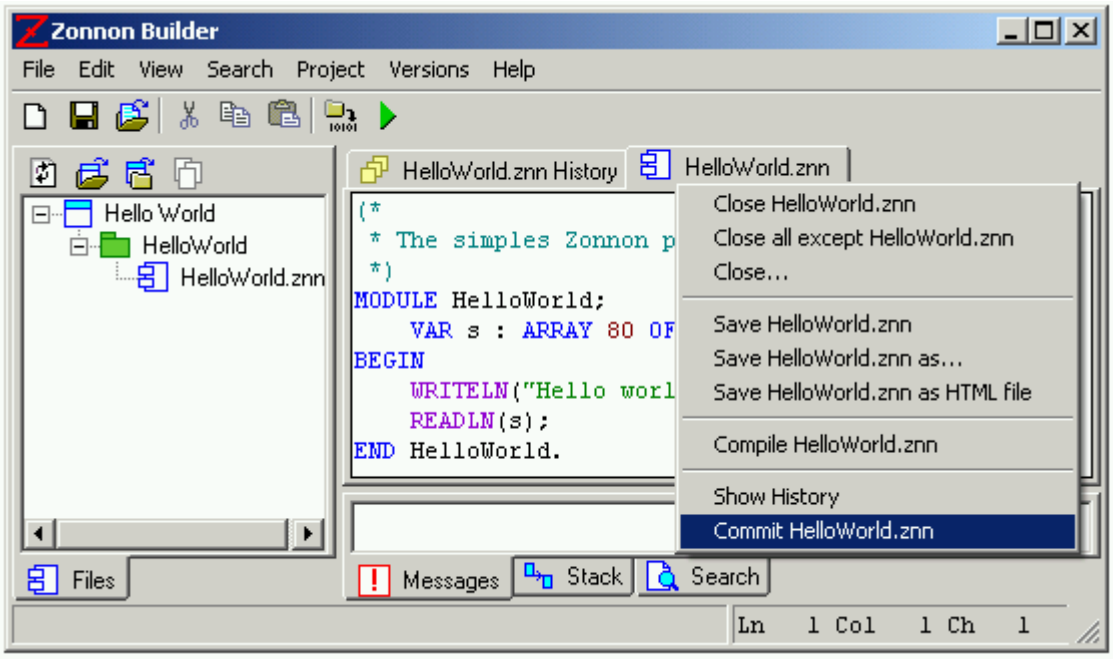


Figure 9.2.8 Selecting **Commit HellowWorld.znn** on the program editor popup menu:

When **Commit** has been selected the following dialog box appears so that a comment can be added the file version to document why the file was committed :

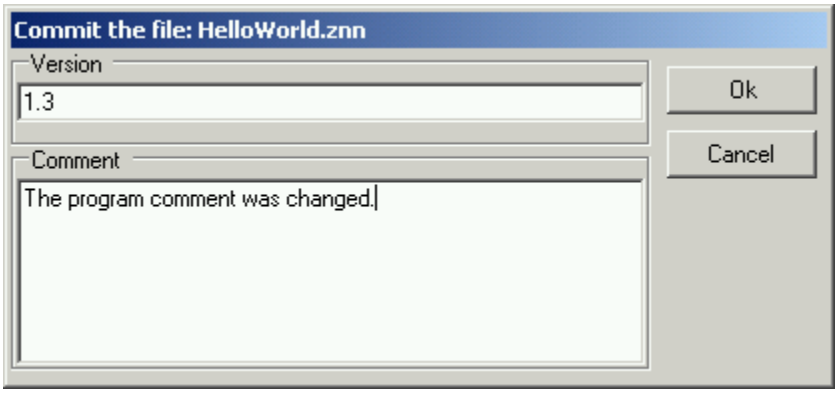


Figure 9.2.9 Commenting in committed file version.

The next file version number will be suggested, **1.3** in this case. It is also possible set your own value for the version number, for example **2.0**.

9.3 Project versions

To create project version select the **Commit Project** menu item in **Versions** menu :

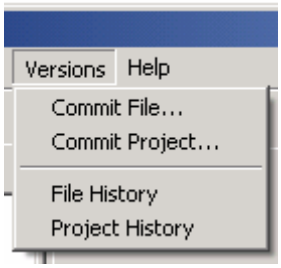


Figure. 9.3.1. The **Versions | Commit Project** menu item on the main menu

When **Commit Project** has been selected the following dialog box appears so that a comment and a label can be added the project version to document why the project was committed :

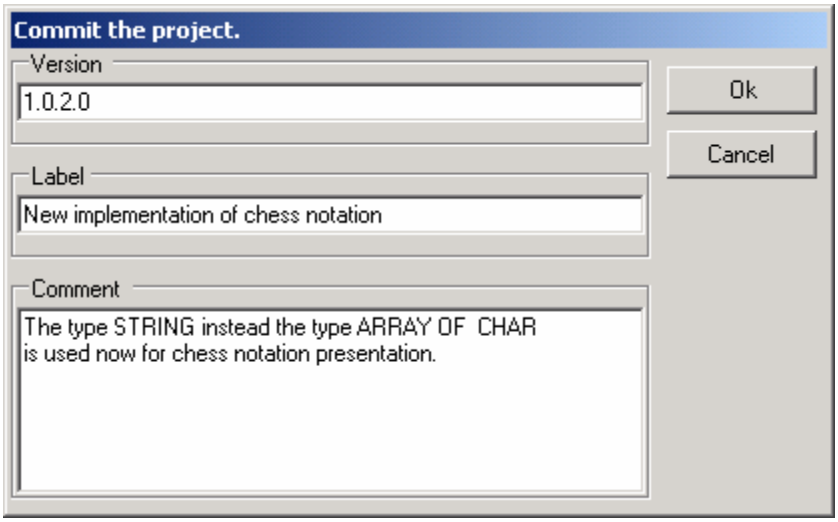


Figure. 9.3.2. Commenting in committed project version.

Lets look at the project history now. Select the **Project History** menu item in **Versions** menu. :

The project history window appears in the Zonnon Builder's centre window, it has three tab windows: **Info**, **Contents** and **Diff**.

In the **Info** tab window the top panel contains the project history list, where the top list item presents current project. Current project item has a **Project icon** . The next items are the versions of the project, each project version has a **Project Version icon** .

Every project history list item has a data and a time attribute. Select the project **Version 1.0.2.0** in the project history list. The project label and comment will be shown at bottom of **Info** tab window :

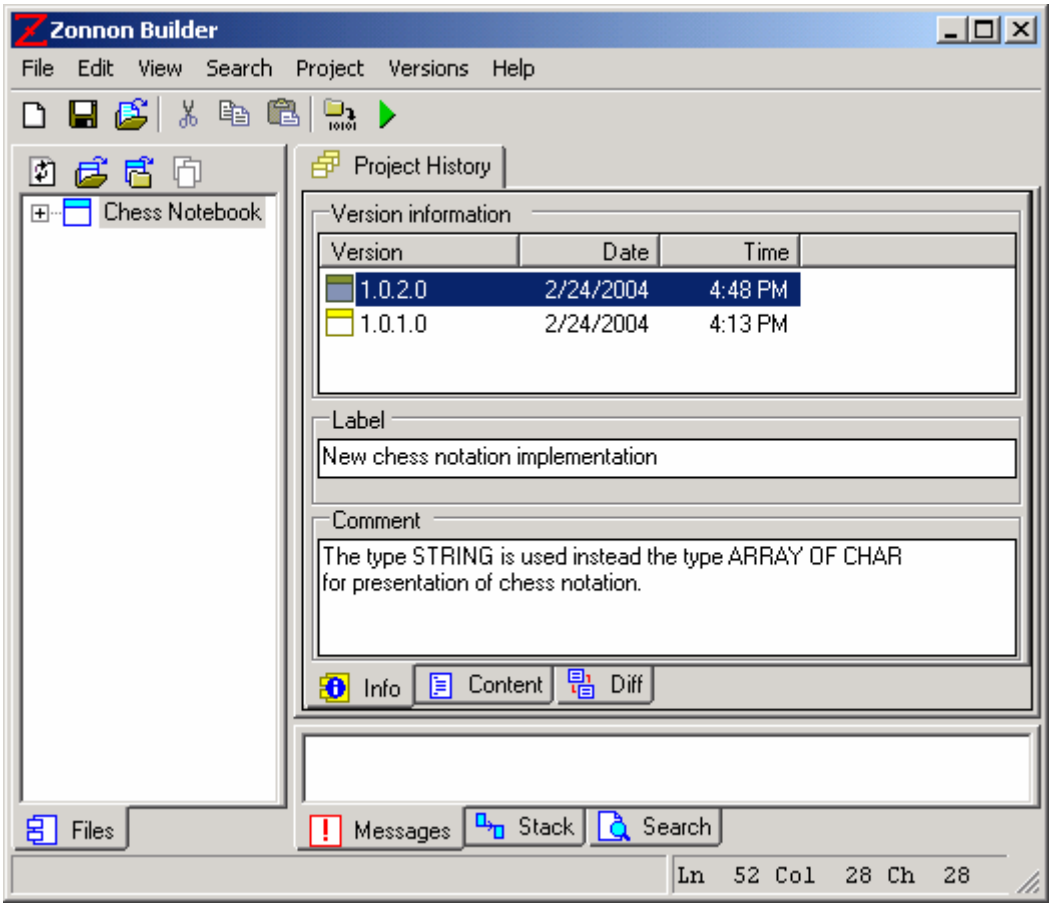


Figure. 9.3.3. The information for **Version 1.0.2.0** of **Chess Notebook** project.

Now lets look at the **Content** tab window. The top panel contains the project history list including the project versions, whilst the bottom panel contains the list of project version files. Each element of the file list has the file version, the file modification data and time, the file directory.

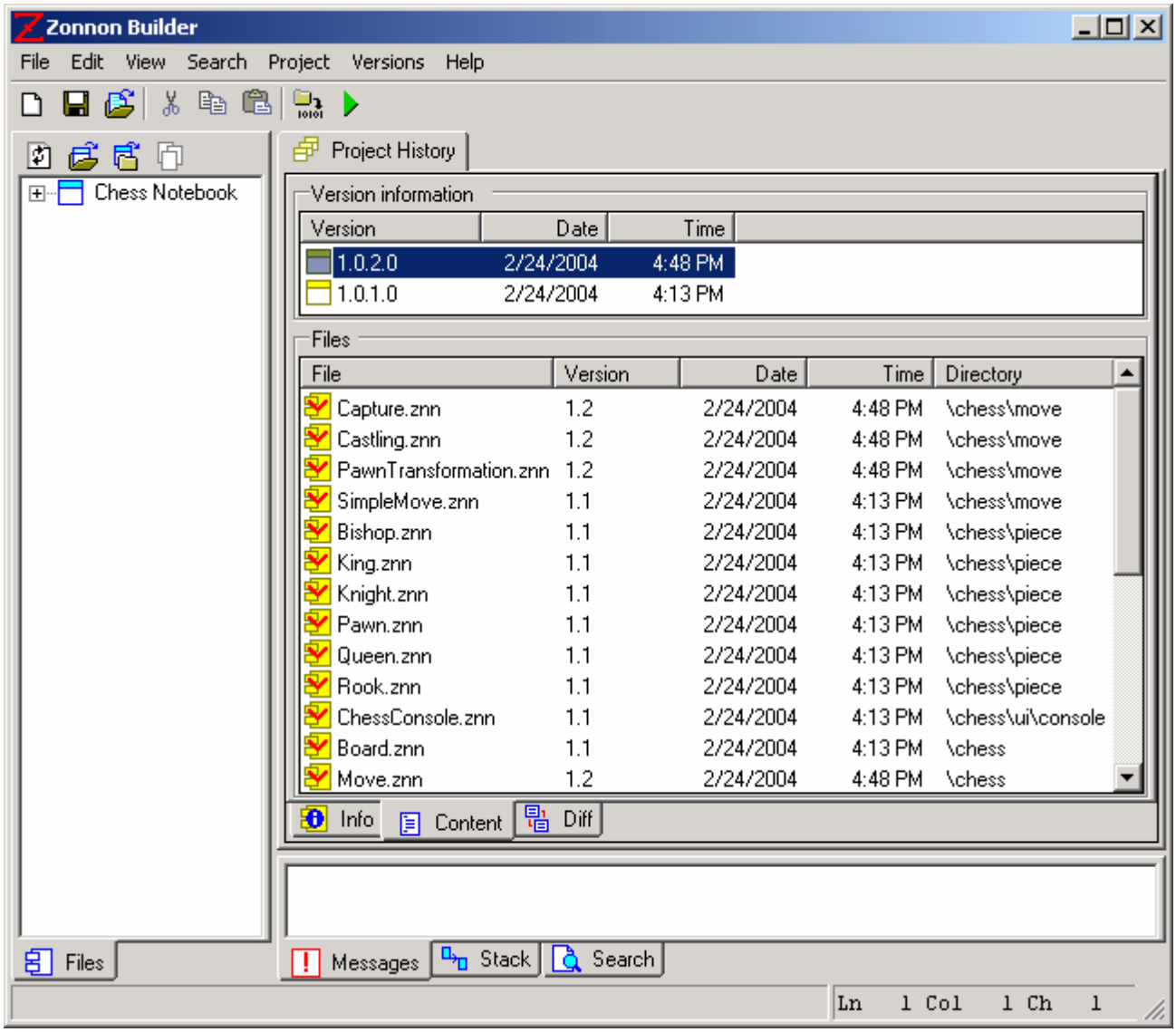


Figure. 9.3.3. The content for **Version 1.0.2.0** of **Chess Notebook** project.

Lets look at the **Diff** tab window. The top panel contains two colored project history lists which make it easy to compare the content history of the project in the lists. The differences between the project's versions is shown in the lower panel.

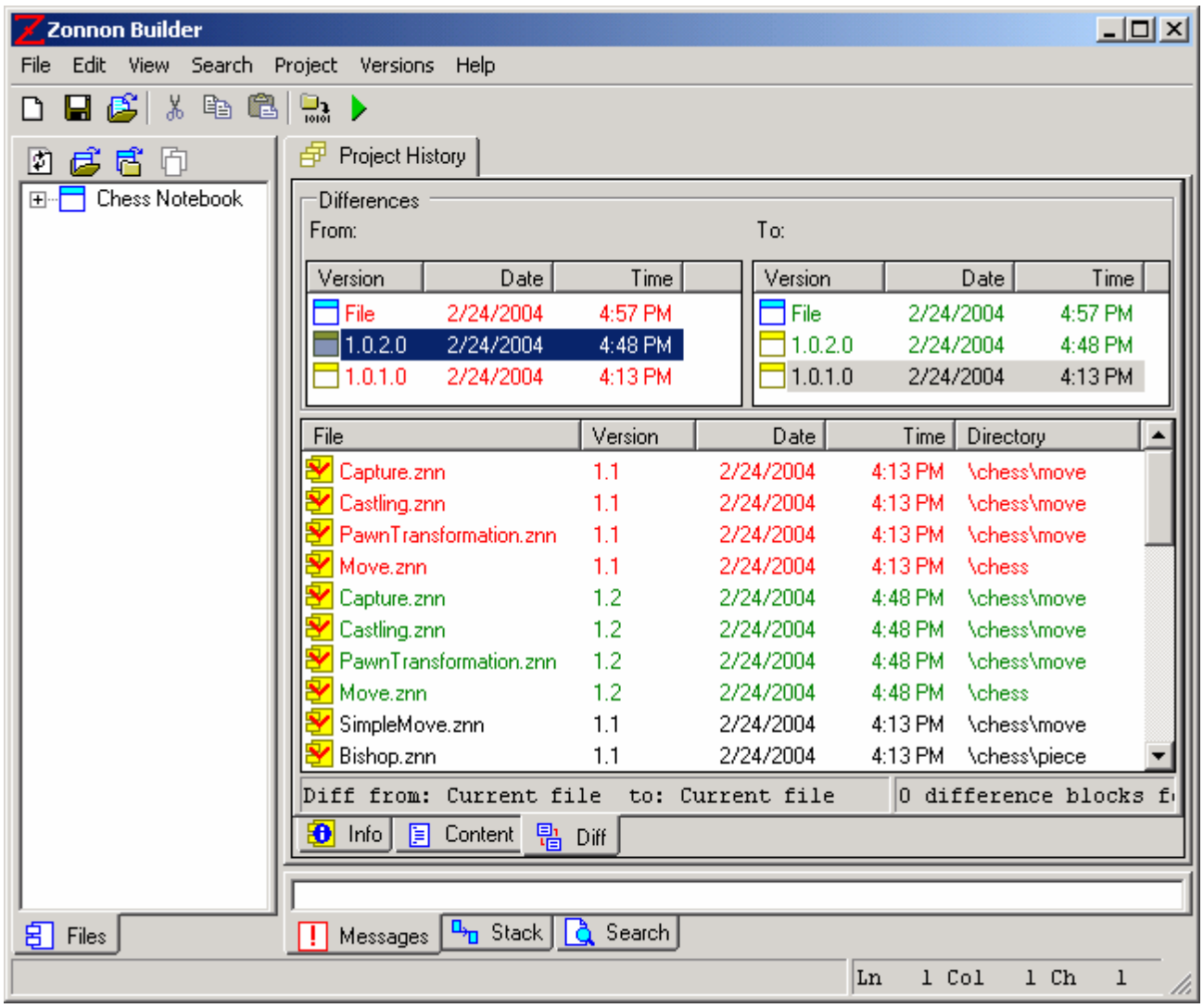


Figure 9.3.4 The differences between **Version 1.0.1.0** and **Version 1.0.2.0** of the project.

If the files exist in both project versions and have same version number then the difference list single element has black color. Else the different file versions are presented in the difference list with thier project colors.

10. Editor Options

10.1 Program representation options.

The program editor supports a number of character attributes which can be set up to change the appearance of the text. Program text contains the following lexical elements: keywords, numbers, strings, comments and predefined language identifiers. Each of these elements has attributes associated with each of its constituent characters: they are normal, bold and italic fonts and the color. All program text also has attributes for font family and font size. To set up your own values of these attributes select menu item **Edit | Editor Options...** as shown below :

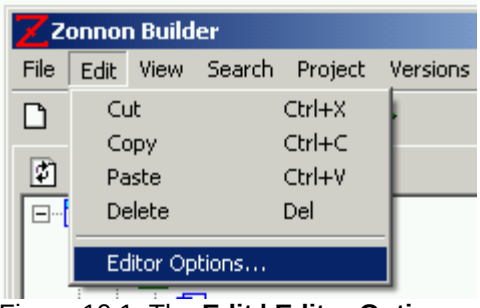


Figure 10.1 The **Edit | Editor Options...** menu item on the main menu.

The editor options dialog box will appear :

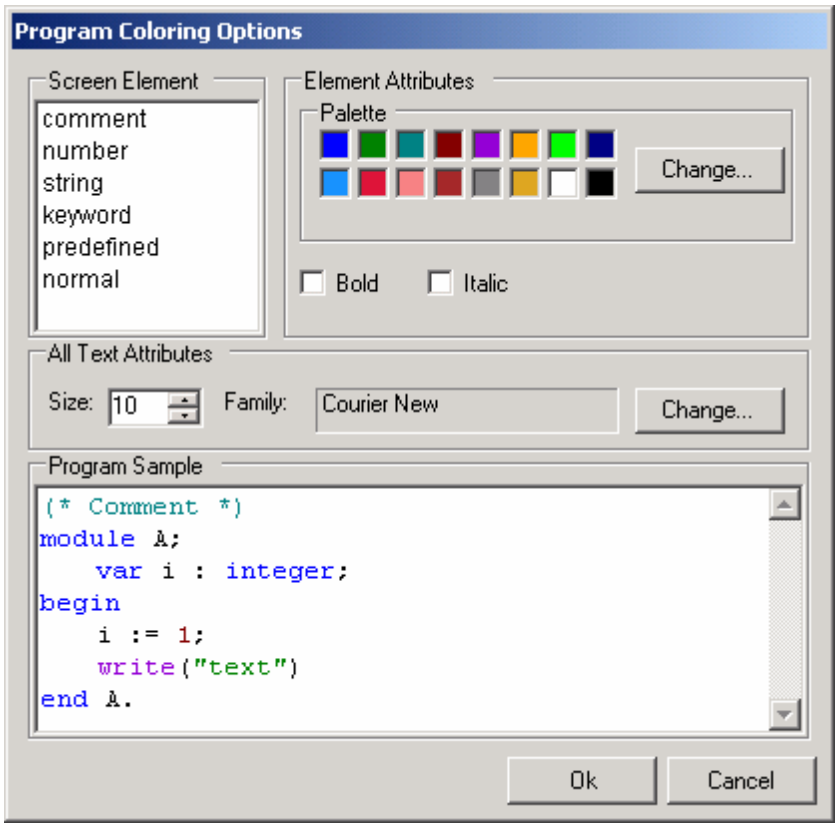


Figure 10.2 The editor options dialog box.

Lets change the presentation of the keyword in the program editor. The default keyword presentation is blue and in plain text. To set the new presentation click on the red square in the **Palette** controls group and check the **Bold** check box. The new representation of the keywords will shown in the **Program Sample** panel:

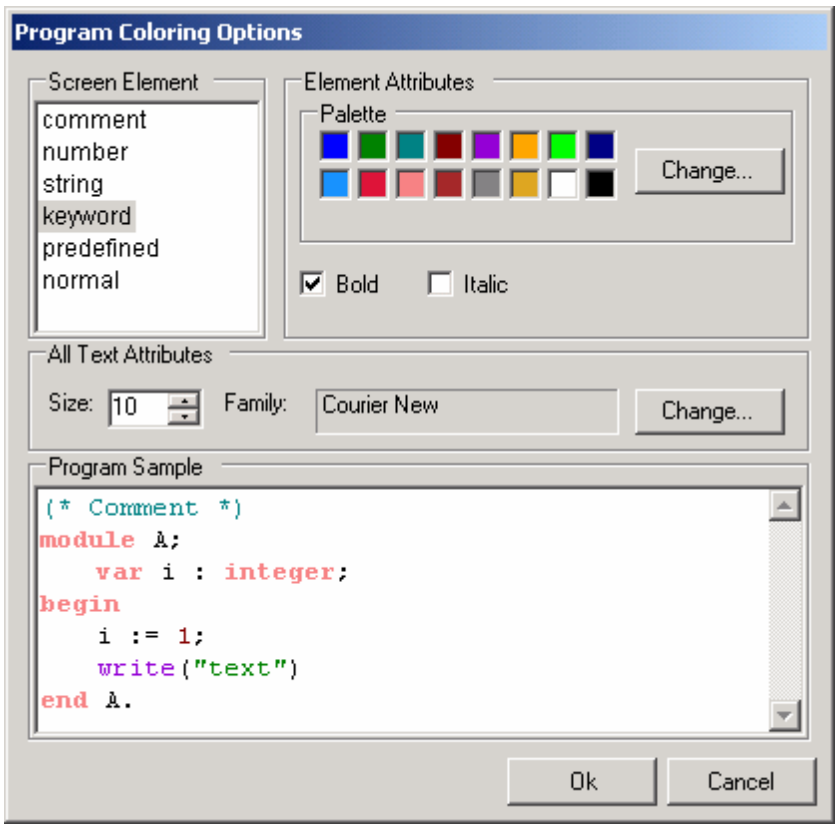


Figure 10.3 The new keywords presentation.

The colors in **Palette** can also be changed by pressing the **Change...** button in the Palette controls group. The standard dialog box for color editing will appear with the available Palette colors being presented in **Custom Colors** panel. It is now possible to edit the content of each palette element to create your own custom palette.

10.2 Program text options.

Lets now change the program text font size. Use the *NumericUpDown* control element with name **size** to do this. The new font size will now used for all characters in the text which belong to the currently selected kind of symbol :

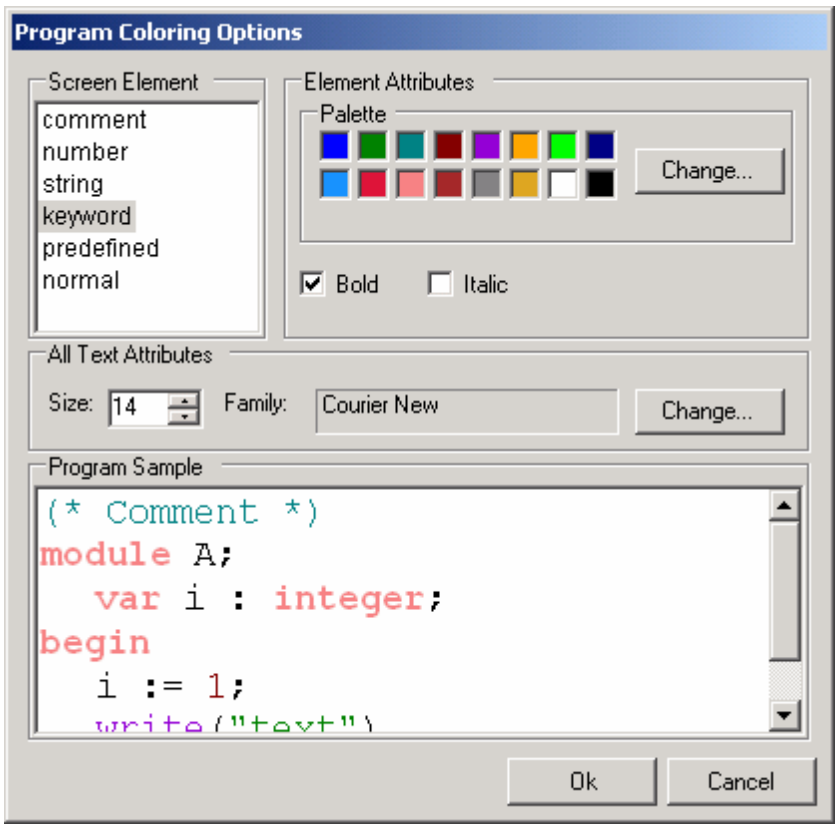


Figure 10.4 The new program font size.

Lets change the program text font size. Press the button **Change...** in the controls group All text Attributes. The standard font editing dialog box will appear:

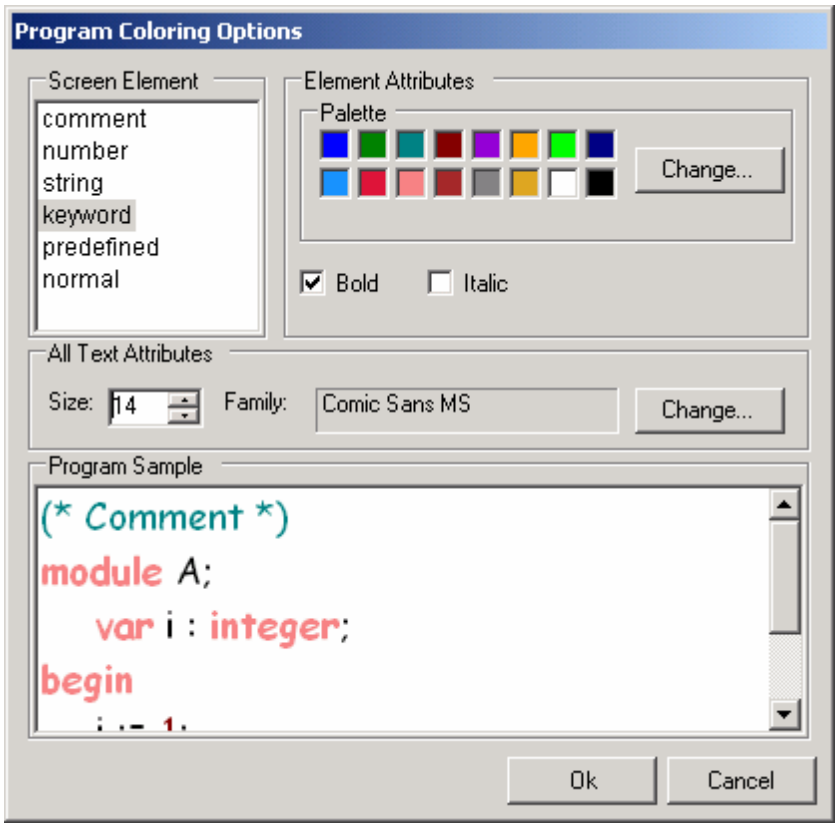


Figure 10.5 The standard font editing dialog box.

Select a font family and/or font size then press OK button. The program text with the new font will appear in the **Program Sample** panel:

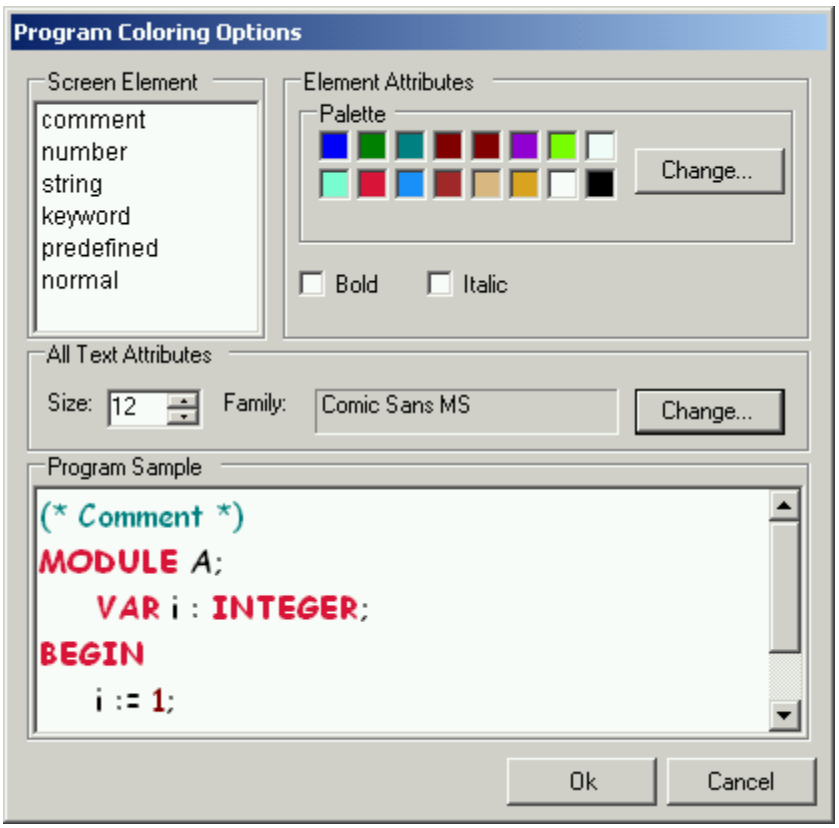


Figure 10.6 The new program font size and family presentation.

Now press the OK button and open the program file **HelloWorld.znn**. The program text with new editor options will now appear using the new attribute values :

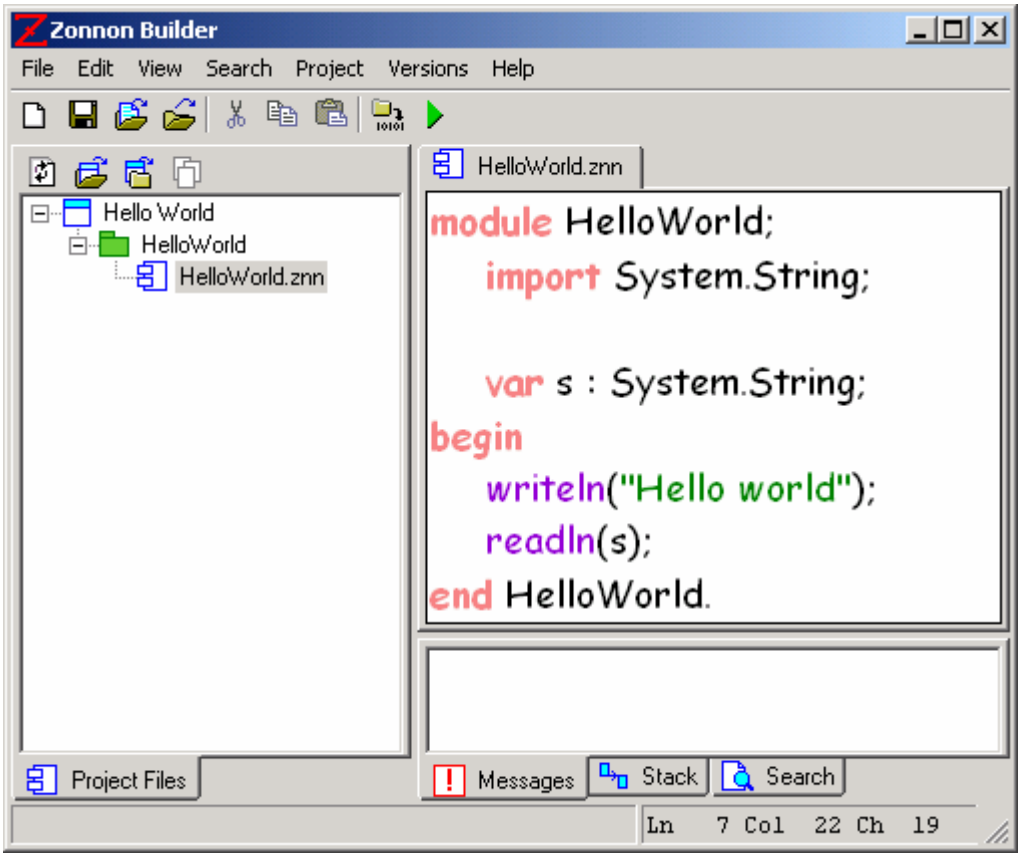


Figure 10.7 The colored program text with new editor options.

Links and Feedback

1. The Zonnon project site at ETH Zurich: http://www.zonnon.ethz.ch
2. The Zonnon Builder section of the site: http://www.zonnon.ethz.ch/compiler/zb

If you have any feedback on this document or on the Zonnon Builder itself then please email them to zonnon@inf.ethz.ch.