

VisualAge C++ Professional for OS/2 and Windows NT

Integrated Development Environment Concepts and Tasks

Version 4.0



VisualAge C++ Professional for OS/2 and Windows NT

Integrated Development Environment Concepts and Tasks

Version 4.0

First Edition (September 1998)

This edition applies to Release 4.0 of IBM VisualAge C++ Professional and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product. The term ″VisualAge″ as used in this publication, refers to the VisualAge C++ product set.

If you have comments about this document, address them to:

IBM Canada Ltd. Laboratory
Information Development
2G / 345 / 1150 / TOR
1150 Eglinton Avenue East
North York, Ontario, Canada, M3C 1H7
FAX: (416) 448-6161
torrcf@ca.ibm.com

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1998. All rights reserved.
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices . . . . . **v**
Programming Interface Information . . . vi
Trademarks and Service Marks vi

Chapter 1. Integrated Development Environment (IDE) **1**
Links Between Panes 2
Projects 4
Workbook 5
Configuration Files 8

Chapter 2. Create or Modify a Project **11**
Create a Project 11
Open or Close a Project 11
View a Build Setup (Configuration File) 12
Change a Build Setup (Configuration File) 13
Configure Change Control Support. . . . 13

Chapter 3. Browse Your Project **15**
Search in the IDE 15
Search Using a Search View 16
Search a View Using Live Find 17
Filter a View 17
Add or Delete a Filter 18
Modify a Filter 18
Add or Delete a Bookmark 19

Chapter 4. Editing **21**
Editor Personalities 22
Editor Commands and Parameters 23
Parsers 23
Creating a New Parser 23
Setting Up Live Parsing 24
Deciding on a Load Macro 24
Macros and Profiles 24
Standard Editor Profiles 24

Chapter 5. Work with Text in the Editor **27**
Enter and Edit Text 27
Undo Changes 29

Chapter 6. Mark and Manipulate Blocks of Text **31**
Mark Blocks of Text 31
Unmark a Block of Text 32
Manipulate Blocks of Text 32
Change the Default Marking Mode. . . . 33
Using Location Marks in a File 34

Chapter 7. Find Text or Locations in a File **37**
Find a Specific Line in a File 37
Find Text 37
Find and Replace Text 38

Chapter 8. Manage Files in the Editor **39**
Create a New File 39
Open an Existing File 40
Save a File 41
Embed Another File into the Current Document 41
Work with Multiple Documents 42
Set Breakpoints in Program Source Files 43
Issue Editor Commands 44
Issue Multiple Editor Commands 44

Chapter 9. Customize the Editor Appearance and Function **47**
Enable the Annotation Column 47
Change Editor Tab Settings 47
Change the Base Editor Font 48
Changing Token Display Attributes . . . 48
Customize the Keyboard 49
Change Text and Background Color Palettes 51
Use Profiles to Modify Editor Behavior 52
Change Editor Load Profiles 54
Write Macros for the Editor 54

© Copyright IBM Corp. 1998

Notices

Note to U.S. Government Users -- Documentation related to restricted rights -- Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used.

Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Director of Licensing
Intellectual Property and & Licensing
International Business Machines Corporation
North Castle Drive, MD - NC119
Armonk, New York 10504-1785
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Canada Ltd., Department 071, 1150 Eglinton Avenue East, North York, Ontario M3C 1H7, Canada. Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

This publication may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

IBM may change this publication, the product described herein, or both.

© Copyright IBM Corp. 1998

Programming Interface Information

Programming interface information is intended to help you create application software using this program.

General-use programming interfaces allow the customer to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification, and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and Service Marks

The following terms are trademarks of the International Business Machines Corporation in the United States or other countries or both:

AIX
AS/400
DB2
CICS
IBM
OS/2
OS/390
OS/400
RS/6000
San Francisco
VisualAge
Visual Servlet
WebSphere

Lotus, Lotus Notes eSuite, and Domino are trademarks or registered trademarks of the Lotus Development Corporation in the U.S. and/or other countries.

Tivoli Management Environment, TME 10, and Tivoli Module Designer are trademarks of Tivoli Systems, Inc.

Encina and DCE Encina Lightweight Client are trademarks of Transarc Corporation.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks or registered trademarks of Microsoft Corporation in the U.S. and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and/or other countries.

UNIX is a registered trademark in the U.S. and other countries licensed exclusively through X/Open Company Limited.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks or registered trademarks of Intel Corporation in the U.S. and/or other countries.

Other company, product, and service names, which may be denoted by a double asterisk(**), may be trademarks or service marks of others.

Chapter 1. Integrated Development Environment (IDE)

The VisualAge C++ Integrated Development Environment (IDE) provides a single, customizable interface in which you can edit, browse, compile, and debug your applications.

The IDE is an object-based environment. You view and manipulate objects, such as classes, source files, and functions, as you work on your application. A colored symbol with a letter in it, such as , identifies each object's type.

You work with one project at a time in the *workbook* area of the IDE. The workbook has pages that are set up for various tasks that you can perform on an object. Pages are grouped into tabbed sections. Each page is broken into one or more panes to further refine presentation of the information about the object. The panes in a page are linked, so that objects flow from one pane to another.

The VisualAge C++ IDE has *toolbars* that provide actions and access to dialogs, so you can easily work with the objects. You can fully customize both the content and position of the toolbars to meet your needs.

The IDE also has an *information area* that displays the build status and other temporary status messages. It also displays help on menu items, and contains an activity meter. With a glance you can see the outcome of the build, and whether you have made any changes to the project since the last build.

Menus offer you many ways to get your work done. The main *menu bar* can be accessed with a mouse or by using keyboard shortcuts. There are also pane, object, and view menus on each pane's title bar. You can open *pop-up menus* for objects in the IDE by clicking the right mouse button with the pointer over an object. You can also click mouse button 2 on a view background to display its pop-up menu. The pop-up menu choices depend on what is beneath the pointer when you click the mouse button.

The IDE uses the information in a *codestore,* a database that contains a representation of the code in a project, to populate the views in the workbook. A build updates the codestore, and refreshes the IDE views.

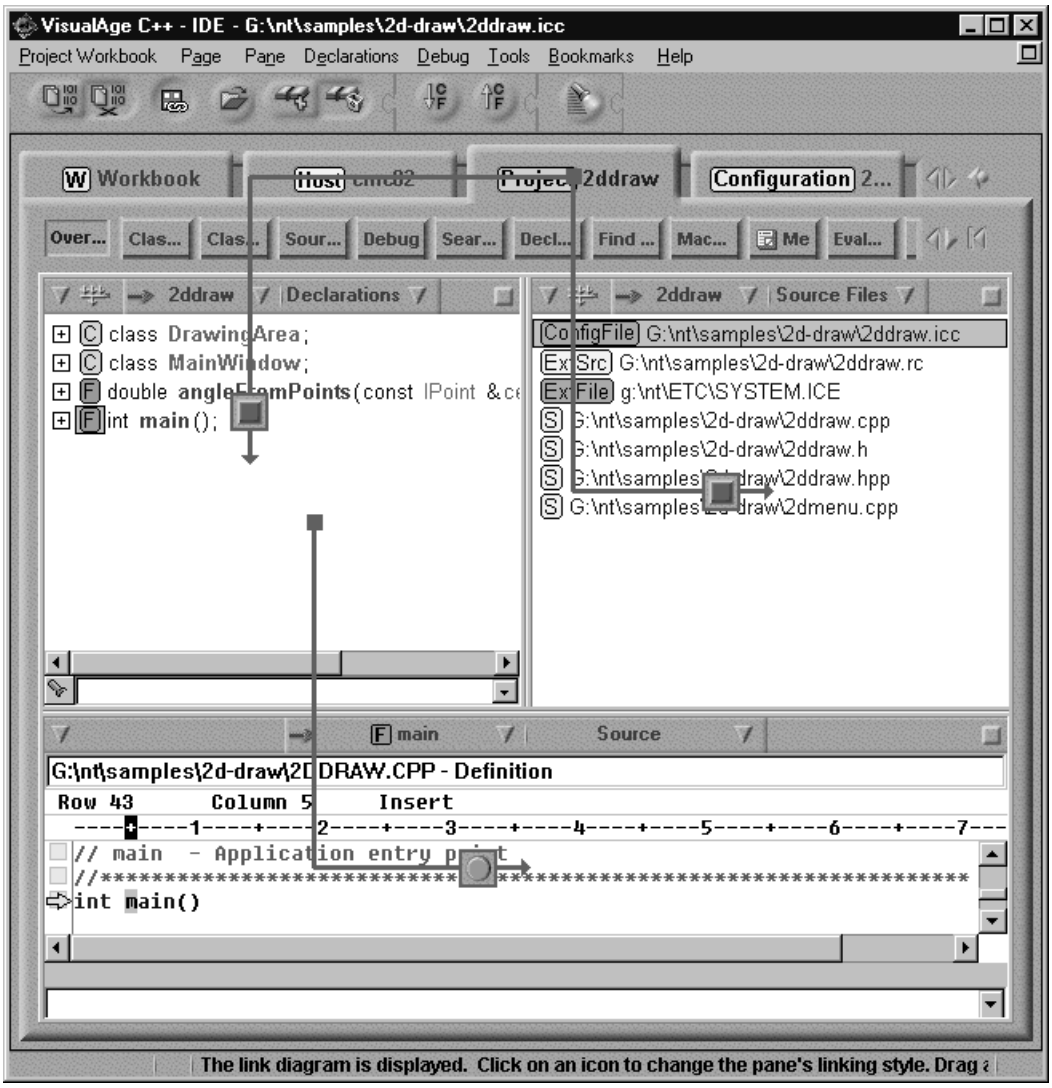
The IDE provides access to a number of tools, like migration tools and optimizers. From the IDE, you can also move source files in and out of the change control system where you control your source.

© Copyright IBM Corp. 1998

Links Between Panes

One of the IDE's key features is the flow of objects between panes. The object on the tab flows into one or more panes on a page, and from these panes, objects can flow to other panes. In each pane, the view displays information about the object. As objects are selected in a view, these objects flow out of the view to other panes on the same page. The flow of objects occurs on *links*. Each pane must have at least one input link, and at least one pane on a page must be linked to the object on its section tab. Without this input, there is no information displayed.

You can open a link diagram, as shown in the image below, to display the links between panes and to the tab. The diagram overlays the current page, and temporarily disables actions on the page. You can use the link diagram to view and change the link structure. Your ability to work with actions on the page is restored once the link diagram is hidden.



Links allow information to flow in only one direction. There are two kinds, manual and automatic . *Manual links* link two panes or a pane to a tab together without regard for what pane has focus. The link remains the same until you modify it. The source of an *automatic link*, on the other hand, is whatever pane has focus. In the image above, the Source view has automatic input link. The pane containing the Declarations view has focus, so the source for the automatic link is this pane. The manual links from tab to the panes containing the Declarations and Source Files views will not change because of a change in focus.

If a pane that has an automatic input link is itself selected, then the link remains on the pane previously selected. If the pane containing the Source view is selected next, the source of the link for this would would still be the pane containing the Declarations view since a link cannot start and end in the same pane. If you want a pane not to have an effect on other panes with automatic input links, that pane can be set to reject automatic linking. In the image above, if the pane containing the Declarations view did not accept automatic linking, then the pane for the Source view would not be affected by selections in the Declarations view.

The flow of objects over a link can be temporarily stopped by detaching a link. The link between the panes remains, and the flow of objects over the links resumes once the link is attached again. When the input link to a pane is detached, the view is unaffected by selections in the other views on the page. Selecting an object in the pane's view, though, still causes that object to flow across any output links, so other views can be affected. In the image above, if you detach the input link for the pane containing the Source view, selecting objects in the Declarations view will no longer affect the Source view.

Links can be customized to meet your needs. You can change the source of a link as well as its type.

Projects

A VisualAge C++ project groups together the source files and actions needed to produce an application or part of an application. To work with your application in the IDE, it must be defined as a project.

A project is composed of the following files:

- a configuration file
- one or more source files
- a codestore

The *configuration file* defines the source files and targets for the project, and defines how those files are to be processed. To be part of a project and its build, a source file needs to be specified in the configuration file, or included by a source file that is specified in the configuration file.

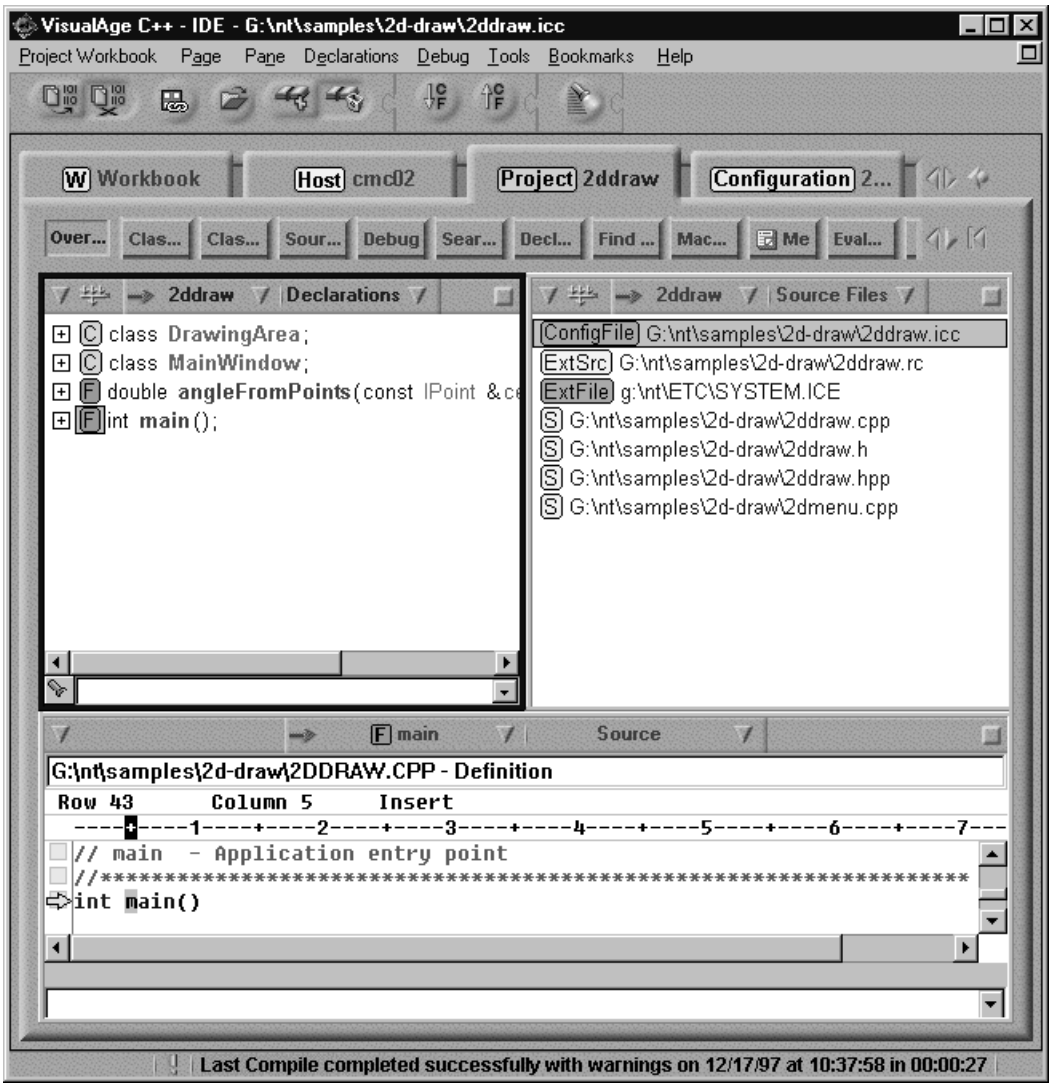
Your source files can be in any directory accessible by VisualAge C++. They can be in one directory or distributed among many. The configuration file's directory is the key directory for the IDE. If a full path name is not used for a source or target file, then the path is relative to the configuration file directory.

In the IDE, you open and work with a single project at a time. The IDE uses the information about your application stored in the codestore to provide many views in the IDE.

A VisualAge C++ project can contain more than C++ source files. C source and resource files are examples of other source files supported.

Workbook

The workbook is the heart of the VisualAge C++ IDE. It has been carefully organized to make the most common coding tasks easier. Where we have not anticipated your needs, you can extend and customize the workbook to suit your particular approach.



The workbook initially has four *sections*. A section organizes information about an object, such as a class or a function. The object's name appears on the *tab* for the section.

- The **Workbook** section is where you view and change information about the IDE.
- In the **Host** section, you can view and work with files on your file system and add files to your project.
- The **Project** section lets you work with code.
- In the **Configuration** section, you can view and define the build for your project.

These four sections are the IDE defaults, but you can extend the IDE by adding your own sections.

Each section contains one or more pages. A *page* contains specific information about the object on the tab, and is organized for a particular task. There are pages for working with source files, debugging or running code, finding uses of declarations, searching, and more. The name on the page button indicates the page's task. If one of the default pages is not set up the way you need it, you can create a new page, or change an existing one.

When a page is created, it is an instance of a *page description*. There can be multiple instances of a page description in the IDE. When you save the page, a page description is created or modified.

Each page contains one or more *panes* which allow more than one type of information to be displayed about an object. Each pane contains information about the object named in its title bar. The pane with the highlighted border has *focus*, and only one pane on a page can have focus at a time.

The information displayed in a pane is called a *view*. Each view has a name and a type, so each view can only display information about certain types of objects. The name of the view is on the right side of any pane title bar. The view can be changed to display different information about the object, but the type of the object determines what views are available. In a view, as you click on objects, they are *selected*. Selected objects are highlighted by a surrounding box that has a dark background. When you select an object in a pane, the pane has focus.

Actions for the various types of objects in the IDE are available both from menus and from toolbars. There are several menus available in the IDE.

- The IDE's menu bar is organized around the IDE objects. The menu between the**Pane** and **Debug** menus changes dynamically to reflect the view that has focus.
- There are also menus available on each pane's title bar providing actions for the pane and the objects displayed in the view.
- Each object has a *pop-up menu* that is displayed when you click mouse button 2 on the object. This menu contains actions for the object. The object with the displayed menu is highlighted with a dashed outline box if it is not the selected object in the view. The view background also has a pop-up menu.

There are four toolbars available in the IDE, and each provides access to actions for a different aspect of the IDE.

- The **Workbook toolbar** contains actions that are global in scope, such as build.

- Actions for the pane that has focus, including link actions, appear on the **Pane toolbar**.
- The **View toolbar** applies to the view that has focus, and the buttons displayed change according to the type of the view. When a Sourceview has focus, this toolbar contains editing actions.
- The **Process toolbar** contains actions that relate to debugging processes.

You can customize toolbars, or hide them. You can add or remove actions from the toolbar, as well as change the appearance of the buttons.

Configuration Files

A set of source files, input libraries, and processing options that are used to generate one or more targets is called a *configuration*. Each configuration is defined using a *configuration file*. A VisualAge C++ project must have a configuration file.

A configuration file is similar to a makefile, but with some important differences. Configuration files are easier to create and maintain than makefiles, and using them can simplify the build process. Unlike a makefile, you do not need to specify any C++ file-dependency or processing information. A configuration file simply contains definitions of targets, the source files used to build those targets, and the options used to process the source files and targets. Once you tell VisualAge C++ what to build, it figures out how to do the build. Makefiles can still be used with configuration files.

A configuration file consists of a series of statements called directives that can be nested. The most commonly used directives are target, source, group, and option directives:

- The target directive specifies the build's output, such as an executable.
- The source directive specifies the source files for a project.
- You can use the group directive to define a group of source files. The group directive does not specify source files for a project, so the source group still needs to be listed in a source directive. The group directive makes dealing with source files that are processed in the same manner easier than dealing with the files individually.
- The option directive defines one or more build options. If you add a name to an option directive, you define an options group. An options group can be used like a variable to apply options that you commonly use together.

There are other directives, such as if, for, and run.

A build processes the configuration file to create a new codestore or to update an existing one. The processing of each configuration file produces one updated codestore. The codestore contains information about the configuration, and is queried to determine changes from build to build.

The default extension for a configuration file is .icc, but you can use any extension. The configuration file can reside in any directory.

When you create a project using the Project SmartGuide, you create the configuration file. The Project SmartGuide can open the Target SmartGuide to allow you to define targets, such as an executable. With a set of views in the Configuration section of the IDE, you can view, add, change and remove target, group, source and option directives easily. You can also apply options to source files, targets and the project, or remove them. To modify other parts of the configuration file, the Configuration section provides Source views where you can use the IDE's Editor to edit the file directly.

Chapter 2. Create or Modify a Project

Create a Project

To work on your application using the IDE, your source files must be part of a project. To create one, select **Create Project...** from the **Project Workbook** menu. This opens the Project SmartGuide, which creates a configuration file using your input.

The SmartGuide has its own online help, which leads you through the process.

Open or Close a Project

To open a project, select **Open Project** from the Project Workbook menu. The Project Workbook menu also contains up to four of the projects you have worked with most recently.

If you do not select a recently-opened project, the **Open Project** dialog box is displayed. If you open a project while another one is already open in the workbook, the first project is closed before the next one is opened.

In the **Open Project** dialog box, select a file with which to open the project.

- If you select a configuration file (.icc), the IDE looks for a codestore file with the same base name and a file extension of .ics using this search path:
  - configuration file's directory
  - current directory
  - DPATH environment variable

If the codestore file is not found, a new one is created in the configuration file's directory.

- If you select a codestore file with the file extension .ics, the IDE looks for a configuration file with the same base name and a file extension of .icc using this search path:
  - codestore file's directory
  - current directory
  - DPATH environment variable

© Copyright IBM Corp. 1998

If the configuration file is not found, a new one is opened with the same base name as the codestore file and a file extension of .icc. The configuration file is created in the file system the first time it is saved. The new configuration file that is created is initially empty and does not reflect what is in the specified codestore file.

- If you select a file that does not have a .icc or .ics file extension, the IDE assumes that the file is a configuration file. The IDE looks for a codestore file with the same base name and a file extension of .ics using this search path:
  - configuration file's directory
  - current directory
  - DPATH environment variable

  If the codestore file is not found, a new one is created in the configuration file's directory.

To close a project, select **Close Project** from the Project Workbook menu. If the project contains any unsaved project source files, you are asked if you want to save them.

View a Build Setup (Configuration File)

The Configuration section is where you view the configuration file that describes your build. The default Advanced page has two views of the entire configuration file. The Details view is a formatted view of the configuration file. You can expand and collapse sections of the configuration file. The Interpreted Configuration view shows how the configuration file was interpreted as it was processed by the build.

While you view the build setup, you can browse and edit source files in the Configuration section. The Source and Groups page and the Targets page both have Source views. When you select a directive or source specifier object, its definition in the configuration file is displayed in the Source view. From the file name field of the Source view, select the source specifier's file name from the list. The source file is displayed in the Source view for browsing or editing.

Change a Build Setup (Configuration File)

The build setup for your project is defined by a configuration file. The IDE offers two ways to change the configuration file, indirectly using a set of views, or directly using a Source view. You can also use any editor outside of the IDE to update a configuration file.

The Configuration section of the IDE's workbook contains several views where you can change the configuration file. If you make changes to the configuration file using these views, the other configuration views are updated. Other views not related to configuration are updated after a build is performed. The configuration views support changing the source, group, option, and target directives. To work with other configuration language directives, you must directly edit the configuration file.

You can edit configuration files from any Source view. The Source page in the Configuration section contains a Source view which has the configuration file as its input object. If you choose to update the configuration file directly, the configuration views will not be updated until a build is performed, or until you select **Refresh Configuration Views** from the Configuration tab's pop-up menu. The build also updates the views in other sections with the changes you have made to the configuration file.

You can also browse and edit source files in the Configuration section. The Source and Groups page and the Targets page both have Source views. When you select a directive or source specifier object, its definition in the configuration file is displayed in the Source view. For source specifiers, click the arrow in the file name field of the Source view, and select the file name from the list. The source file is displayed in the Source view for browsing or editing.

Configure Change Control Support

Change control support in the IDE can be enabled and customized using the Change Control Configuration dialog box.

To enable and customize change control support:

1. Click **Change Control Configuration** on the Settings page of the Workbook section. The Change Control Configuration dialog box is displayed.
2. Clear the **No Change Control** check box to enable the support.

3. Click **Defaults**. The **Select Change Control System for Defaults** dialog box is displayed. Select a system and click **OK**. The **Checkout Command**, **Checkin Command** and **Extract Command** fields contain the default commands for the chosen change control system. These commands can be modified, and commands from other change control systems can be used.
4. If you want to use relative path names for the files, enter the relative path name in the **Relative Path Name** field. The relative path name is a string added to the change control system's file name when it is stored on the local file system.
5. Click **OK** to complete the customization and close the dialog box. The change control configuration is available immediately for use. Click **Cancel** to close the dialog box without changing the change control configuration.

Chapter 3. Browse Your Project

Browsing your code is easy with the IDE. You do not have to process the code with any special options, nor do you have to use any special tools. Browse your project using the views and pages in the workbook.

The search functions provided by the **Search** view and Live Find help you quickly find objects in your project, especially if the project is large.

Search in the IDE

There are three ways to search in the IDE: using a Search view, using a Live Find search, and using a Find Uses view. Which type of search you choose depends on what you want to search.

With both a Live Find or a Search view search, you enter a search pattern, and set search options. The pattern can be a simple text string, or it can be a regular expression. In a regular expression, you can combine characters and special characters, such as brackets, to extend the search function.

Use a Search view to search the contents of an object. The input object to the Search view is the domain for the search. For example, if the input object to a Search view is a project, all of the files belonging to that project are searched. By changing the input object, you change the search domain. All possible matches for the search pattern and the search options specified are listed in the Search view when the search completes.

Use a Live Find search to search a single view in the IDE. It is a dynamic search, so with each character you type for your search pattern, the selected object in the view changes to the next possible match. The Live Find search helps you quickly find text in a view with a lot of information.

Use a Find Uses view to find where an object declaration is referenced. The declarations supported are class, variable, typedef, namespace and enumeration. For this type of search, the Find Uses view is more precise than a Live Find search or the Search view. If you use Live Find or a Search view to look for an object declaration, then every occurrence of the name of the variable the source files, including references in comments, is displayed.

© Copyright IBM Corp. 1998

Search Using a Search View

The Search view takes the text string or regular expression that you specify, and searches the input object to the view. The input object can be a directory, which gives you the option of searching any subdirectories recursively.

When the search completes, the view contains a list of all possible matches for the specified search string.

To search using the Search view:

1. Specify a pattern in the entry field at the top of the Search view. The pattern can be a simple text string or a regular expression. If you want to use a previous search pattern, click on the arrow beside the entry field to display a history list.
2. Set the search options, if you need to. If the search options are not displayed below the entry field, click the plus sign button to the right of the flashlight button to display them:
   - Select **Case Sensitive** to search for an exact case match of the pattern. If this option is not selected, the case of the pattern is ignored.
   - Select **Show Matches** to have the matches for the pattern displayed in the Search view. When this option is not selected, all lines that do not match the pattern are displayed in the view.
   - Select **Find All** to highlight all matches on a line. If this option is not selected, only the first match on a line is highlighted.
3. Click the flashlight button to start the search. The flashlight button will be replaced by a stop sign button during the search. If you click the stop sign to stop the search, the search stops after completing whatever file it is then searching.
4. When the search is complete, all lines containing matches are displayed in the Search view organized by source file. The matches are grouped by the source file they are located in. Hide the matches for a source file by clicking the minus symbol to the left of the source file name.

The Search page, by default, contains a Source view with an input link to a Search view. If you select a match in the Search view, the corresponding source code is displayed in the Source view for you to browse and edit.

Search a View Using Live Find

Live Find is a dynamic search function used for searching the contents of a view.

To search a view using Live Find:

1. Select **Live Find** from the pop-up menu on the background of a view. The Live Find controls will be displayed at the bottom of the view.
2. Select search options, if you need to. Click the flashlight button to display a menu:
   - Select **Case Sensitive** to search for an exact match of the search string.
   - Select **Search Collapsed Lists** to look for matches in any collapsed lists in the view, if this function is supported in the view.
3. Type your search pattern into the entry field. Your pattern can be a simple text string or a regular expression. With the first character you type, the search starts, and continues with every additional character typed. When a match is found, the flashlight button lights up , and the matched object becomes the selected object in the view. If the flashlight turns red, then the regular expression you specified is incomplete. You must complete the regular expression before searching resumes.
4. Once you type in the full pattern, continue the search using the pattern by selecting **Find Next** or **Find Previous** from the flashlight menu.
5. Press Esc to hide the Live Find controls.

If you want to search using a previous pattern, click the arrow beside the entry field, and select the pattern from the history list.

Filter a View

You can filter a view to reduce the number of objects that appear in that view. A filter indicator on the pane title bar indicates whether or not the view in the pane supports filtering. If the filter indicator has a red dot then there may be some information missing from the view. When there is no red dot, all of the information for the view is displayed.

To change a filter, click the filter indicator to display the filter menu. Select any filter from the menu. The default filter is **My Objects**. To turn filtering off, select the **Show All** filter from the filter menu. The red dot on the filter indicator disappears to indicate no filtering in the view. If you change a filter, subsequent views in that pane that support filtering will use the changed filter.

Add or Delete a Filter

Use the Filters page in the Workbook section to add or delete filters.

To add a filter, you can either create a new filter, or copy one and modify it. To create your own filter, click **New** in the Filters view. An unnamed filter without any filter criteria is created, and added to the user-defined filters list. The filter must be changed to add filter criteria so that it will filter a view.

To copy a filter, select the filter that you want to copy from the Filters view. Click mouse button 2 on the filter, and select **Copy Filter** from the pop-up menu. A copy of the filter is added to the user-defined filters list.

You can delete any filter you have created. Select the filter object, and select **Delete Filter** from its pop-up menu.

Modify a Filter

Use the Filters page in the Workbook section to modify a user-defined filter.

1. In the Filters view, select the filter to be changed. The filter is displayed in the Change Filters view.
2. Type the filter's name and description in the **Name** and **Description** entry fields.
3. Add, insert, or remove a filter criteria:
   - To add a filter criteria, select **Add Criteria** from the view's pop-up menu. The criteria will be added to the end of the list of criteria.
   - To insert a criteria after the selected criteria, select **Insert Criteria** from the pop-up menu of the selected criteria.
   - To remove a filter criteria, select **Remove Criteria** from the pop-menu of the selected criteria.
4. If there is more than one filter criteria, select to either **And** or **Or** the criteria to the previous one.
5. To see the criteria details, click the plus symbol to the left of the criteria.
6. In the **Matching Objects** field, select **Include all** to have the objects matching the criteria included in the view. Select **Exclude all** to have the objects matching the criteria removed from the view.

7. In the **Criteria Type** field, click the arrow and select the type of filter criteria from the menu displayed. The following types of criteria are available:
   - **Class** filters based on the type of class
   - **Name** filters based on the object name
   - **File** filters based on the file name
   - **Object Type** filters based on the object type
   - **Existing Filter** filters based on an existing predefined or user-defined filter that you choose
8. Depending on the type of criteria chosen, either fill in the entry field, or make a selection in **Criteria Definition** field to complete the filter criteria.
9. Repeat the process from step 3 to define or remove more filter criteria.
10. Click **Change** to update the filter. If the changed filter is currently in use, the view using it will be immediately refreshed to reflect the filter changes.

Add or Delete a Bookmark

Bookmarks mark pages in the workbook. Use them to easily display another page, especially if it is in a different section.

To bookmark a page, select the page and then select **Add Bookmark** from the **Bookmarks** menu. The bookmark, named for the section and page it marks, is added to the list of bookmarks contained in the **Bookmarks** menu.

To remove a bookmark, select the bookmarked page and then select **Delete Bookmark** from the **Bookmarks** menu. The **Delete Bookmark** action is only available when a bookmarked page is displayed.

Chapter 4. Editing

The IDE includes an integrated editor, referred to as the Source view, which is usually located in the bottom part of the IDE. The Source view provides editing functions that let you create and modify source files.

Clicking in the Source view area changes IDE focus to the Source view. Actions available in the main menu bar and toolbar change to reflect actions relevant to editing operations.

In addition to basic editing functions, the editor also offers the following features:

- An optional annotation area at the start of each line can display debugging breakpoints or highlight program source code pertaining to objects selected in other IDE views.
- Language parsing uses automatic indenting, color, and text effects to emphasize different parts of your source program, such as C or C++ language keywords, logic structures, comment lines, and arithmetic operators.
- Locations marking facilities let you define *bookmarks* that let you move quickly from one location to another within your source file.
- Elaborate search facilities let you specify the precise scope of the search. In addition to locating specific character strings, editor search facilities will also accept regular expression search strings that can finding matching search *patterns* in your source file.
- Multiple editing views let you have more than one file open for editing at a time.
- Multiple editing windows let you open multiple views of the same file for editing, letting you view different parts of the file at the same time. Changes to the file in any view are reflected immediately in other Source views containing that same file.
- An Editor programming language lets you program elaborate macros to automate recurring editing tasks.
- Keystroke recording facilities let you *record* sets of keystrokes, which you can later replay or incorporate into your own macros.

Some IDE functions will open an Editor window outside the IDE, as will some editor commands. The editor can function on its own as a full-featured programming editor independent of the IDE, but you should avoid using the editor in this way for project-related work. Work performed in the editor outside the IDE may not be known or recognized by projects defined within

© Copyright IBM Corp. 1998

the IDE. If an IDE function opens an Editor window outside the IDE, you should use that window only for the specific task for which it was opened, and then close the Editor window immediately.

The Editor window has a different set of controls and performs some functions differently than the Source view. Refer to Editor window contextual help for assistance.

Editor Personalities

You can configure the editor to adopt the keyboard and command personalities of many popular editors.

Most editor personalities differ only in the keys and commands used to perform various editor tasks. Some editor personalities supported by the editor, listed below, also add a prefix information and command area at the start of each line.

Editor Personality	Profile Name
ISPF	ISPFPRF.LX
SEU	SEUPRF.LX
XEDIT	XEDIPRF.LX

The Editor recognizes prefix commands used by these editor personalities. Depending on which profile you are using, you can enter SEU, XEDIT, or ISPF commands when the prefix area is active. You can also create your own commands by binding any editor macro or command to your own line commands.

The prefix area appears on the left side of the Source view or the Editor window, and may display line numbers and/or the date. Line numbers are maintained by the Editor. If the Annotation Column is activated in the View Settings for the Source view, the prefix area will be located next to the annotations column. Refer to Related Tasks below for more information on using prefix area commands for specific editor personalities.

Editor Commands and Parameters

The Editor is fully programmable through the use of an extensive command and parameter set. You can use commands and parameters to customize your Source view or Editor window, search for or change text in your document, or perform many other functions. You can find details about the commands and parameters and their syntax in the Reference section of this online help information.

Parsers

A parser is an Editor command that acts interactively on a file to improve the presentation of the data in the file. A parser uses indentation, colors, and fonts to highlight different items in a programming language file. For example, language keywords are highlighted in one color, variable names in another, and string literals in yet another. Indentation is used to highlight program logic structure.

The Source view or the Editor window displays a file according to instructions given by the parser and associated load macros. You can add your own instructions to:

- Change the special keys, fonts, and view for each file type by customizing a load macro. In the Editor window you can also change the Actions Menu.
- Write your own parser to operate on other types of files.
- Add your own list of keywords and associated fonts to the parsers that come with the Editor.

Creating a New Parser

To create a parser, you need programming skills, preferably in C. You also need to be familiar with creating external commands. You also need to understand the concepts of lines, fonts, and classes.

A parser is an external command that can be executed from the command line or from within macros. Commands for parsers are added to the Source view or the Editor window by writing Editor extensions using the Applications Programming Interface (API). Because parsers are called quite often, and may have to perform a lot of processing, they are usually written in the C language.

Setting Up Live Parsing

Live parsing means that the parser monitors and records all changes made to the file as you make them. New data is colored as you add it.

Deciding on a Load Macro

A parser normally works with one particular type of file, such as C or COBOL. The file type is indicated by the file-name extension. Each time a file with a specific file-name extension is loaded, a load macro is called to set up the instructions for the parser. Typically, the load macro does the following:

- Sets fonts and in the Editor window you can change the Actions keys and a special menu.
- Invokes an initial first-pass parse of the whole file.
- Sets up the live parser command.

You can use the load macros supplied with the Source view or the Editor window as examples when developing your own load macros.

Macros and Profiles

Much of the Editor's flexibility come from profiles, which are text files containing Editor commands. Profiles are a specialized form of Editor macro files. Some of the profiles supplied with the Editor provide specific editing features and run automatically at specific times. You can also define your own profiles and macros and run them at any time you want.

Standard Editor Profiles

The standard Editor profiles are kept in the **macros** directory.

Note: AIX

Uses the **/usr/vatools/macros** directory.
OS/2

WIN

Uses the *product_installation***macros** directory.

You can edit these profiles directly, but if you change them incorrectly, the behavior of the Editor may be affected. Instead, if you need to modify the behavior of the Editor, do so by defining load profiles.

The following types of profiles are provided with the Editor:

xxx.LXS	This profile runs whenever a file with filename type *xxx* is saved, including autosaves. Use of this profile is optional. For example, you could create an CPP.LXS profile to be run when files with a CPP source type are saved.
xxx.LXL	This profile runs whenever a file with filename type *xxx* is loaded, or opened for editing. Use of this profile is optional. For example, you could create an CPP.LXS profile to be run when files with a CPP source type are opened for editing.
PROFILE.LX	This is a profile in which you can save your own customization commands. It is the last standard profile that is run before a file is opened, and its commands override the commands used in previous profiles.

Chapter 5. Work with Text in the Editor

Enter and Edit Text

Entering Text

- Click in the working area of the Source view or the Editor window and begin typing.
- To type on the command line, click in the command line area or press the Esc key and then begin typing.

Replacing and Inserting Text

- The status line indicates which mode the Editor is in. When the Editor is in *Replace* mode, the cursor appears as a solid block, one character width in size. Text overlaid by this block will be replaced by any new text that you type. When the Editor is in *Insert* mode, the cursor appears as a thin vertical line. Any new text that you type is inserted into the file at the cursor position, and existing text to the right of the cursor will be shifted to the right.
- To toggle back and forth between the two modes, press the Insert key.

Deleting Text

- There are many ways to delete unwanted text. To delete single characters, do one of the following:
  - Press the Delete key. If you are in *Insert* mode, the character to the right of the cursor is deleted; if you are in *Replace* mode, the character that is overlaid by the cursor is deleted. Text to the left of the cursor moves right to fill the resulting gap.
  - Press the Backspace key. The character to the left of the cursor is deleted. Text to the right of or overlaid by the cursor moves left to fill the resulting gap.
- To delete all the text between the cursor and the end of the current line, press Ctrl+Delete.
- To delete a complete line, move the cursor to the unwanted line and press Ctrl+Backspace.

Moving the Cursor

- Position the mouse pointer where you want the cursor to be, and click mouse button 1, or,
- Press the Up, Down, Left, or Right arrow keys, or,

© Copyright IBM Corp. 1998

- Press the Home key to move the cursor to the beginning of a line, or the End key to move it to the end of a line, or,
- Press Alt+Left arrow to move the cursor one word left, or Alt+ Right arrow to move it one word right, or,
- Press the Page Up or Page Down keys to move the cursor up or down one window at a time, or,
- Press Ctrl+Home to move the cursor to the beginning of the file, or Ctrl+End to move it to the end of the file, or,
- Press Ctrl+J to return the cursor to the place in the file where you last entered text.

  Tip! If you inadvertently select text by dragging the mouse, deselect it by pressing Alt+U, or, click the right mouse button and choose **Deselect all** *from the pop-up menu.*

Inserting a Blank Line

- Move the cursor to any point on a line and press Ctrl+Enter, or,
- Move the cursor to the end of a line and press Enter.

  A blank line is created after the current line, and the cursor moves to the first column position of this new line.

Splitting and Joining Lines of Text

- To split a line:
  - Place the cursor where you want the break to occur, and press either Alt+S or Enter. Pressing Alt+S leaves the cursor at its current position; pressing Enter moves the cursor to the beginning of the new line.

  When you split a line, all the text to the right of the cursor is moved down to a new line.
- To join two lines, do one of the following:
  - If the cursor is at the end of a line, press the Delete key to join the current line with the next line.
  - If the cursor is at the beginning of a line, press the Backspace key to join the current line with the previous line.
  - With the cursor anywhere on a line, press Alt+J to join the current line with the next line together.

Undo Changes

The Editor records each set of changes you make to a file in the Source view or the Editor window. The number of changes made since the last file save is displayed on the status line.

If you want to undo a set of changes to a file, do the following:

1. Select **Source** from the main menu bar in the Source view, or **Edit** from the Editor window menu bar.
2. Select **Undo** from the resulting pull-down menu. The last set of changes to the file is undone.
3. Repeat the above steps until all unwanted changes are undone.

You can also cancel the effects of an undo operation by doing the following.

1. Select **Source** from the main menu bar in the Source view, or **Edit** from the Editor window menu bar.
2. Select **Redo** from the resulting pull-down menu. Changes made to the file by the most recent undo operation are cancelled.
3. Repeat the above steps until all the effects of all unwanted undo operations are cancelled.

Chapter 6. Mark and Manipulate Blocks of Text

The editor offers facilities that let you mark blocks of text in various ways to accomplish different results. Once you have marked a block of text, you can manipulate the marked block with either standard operating-system clipboard facilities or the Editor's own block manipulation facilities.

Mark Blocks of Text

There are many ways of marking text in the Source view or Editor windows. Some are described below.

Note: Some instructions below mention selecting **Source** from the main menu bar. If you are working in the Editor window, you must instead select **Edit** from the Editor main menu bar.

Marking a Block of Text with the Mouse

1. Position the mouse pointer over the character that is to start the block.
2. Press and hold mouse button 1.
3. Drag the mouse pointer to the to the end of the block of text you want and release the mouse button. The block is now marked.

   Note: The default block marking mode is used when marking text with the mouse.

Marking a Block of Text using the Keyboard

1. Use the cursor keys to move the cursor to the character that will start the block.
2. Press Alt+B.
3. Move the cursor to the end of the block of text.
4. Press Alt+B again. Text between the two points is marked.

Marking an Entire Line of Text

© Copyright IBM Corp. 1998

- Select **Source** from the main menu bar, select **Block** from the resulting pull-down menu, and then select **Mark Line** from the cascading menu, or,
- Place the cursor anywhere on the line and press Alt+L, or,
- With the Ctrl key depressed, double-click on the line.

Marking Multiple Lines

1. Place the cursor on the first line to be marked and press Alt+L.
2. Move the cursor to the last line to be marked and press Alt+L again. The two selected lines, and all lines between them, are marked.

Unmark a Block of Text

To unmark a block of text, use any of the following methods:

- Select **Source** from the main menu bar (or select **Edit** in the Editor window), select **Block** from the resulting pull-down menu, and then select **Unmark**from the cascading menu.
- Click the right mouse button and select **Deselect all** from the resulting pop-up menu.
- Press **Alt+U**.

Manipulate Blocks of Text

After you have marked a block of text, you can manipulate it in any of the ways described below.

Note: Some instructions below mention selecting **Source** from the main menu bar. If you are working in the Editor window, you must instead select **Edit** from the Editor main menu bar.

Copying a Marked Block

1. Move the cursor to the location in the file where the blocked text will be copied.
2. Select **Source** from the main menu bar.
3. Select **Block** from the resulting pull-down menu.

4. Select **Copy** from the cascade menu. A copy of the marked block is inserted at the cursor position.

   Alternatively, position the cursor and then press Alt+C to copy a selected block.

Deleting a Marked Block

1. Select **Source** from the main menu bar.
2. Select **Block** from the resulting pull-down menu.
3. Select **Delete** from the cascade menu to delete the marked block.

   Alternatively, press Alt+D to delete a marked block.

Moving a Marked Block

1. Move the cursor to the desired location.
2. Select **Source** from the main menu bar.
3. Select **Block** from the resulting pull-down menu.
4. Select one of the following options:
   a. Select **Move** from the cascade menu. The block is placed below the cursor. Existing text below the cursor shifts downward to allow room for the moved block of text.
   b. Select **Overlay** from the cascade menu. The block is placed below the cursor, overwriting any existing text.

   Alternatively, position the cursor and then press Alt+M, or use Alt+Z if you want to place the block over existing text.

Change the Default Marking Mode

You can change the default marking mode to one of the following choices:

stream	Default. The cursor is coupled with the marked text selection. Marked text will be deselected if you click the mouse at another location in the file.
character	Similar to stream, except that the cursor is *not* coupled with the marked text selection. You can mark a block of text with the mouse, and then click the mouse at another location in the file without unmarking the block of text already selected.
element	Whole elements are marked. The cursor is not coupled to the marked text selection.

rectangle Rectangular areas can be marked starting at any character position on a line and extending over as many lines as needed. The cursor is not coupled to the marked text selection.

To change the default block marking mode for the Source view or the Editor window:

1. Open the **edit command** line by pressing the **Esc** key.
2. Enter the following command,

   ```
   set blockdefaulttype markmode
   ```

   where

   ```
   markmode
   ```

   is one of the modes listed above.

To determine the current default block marking mode:

1. Open the **edit command** line by pressing the **Esc** key.
2. Enter the following command,

   ```
   query blockdefaulttype
   ```

3. The current blocking mode is returned in the message area at the bottom of the Source view or Editor window.

Using Location Marks in a File

Location marks function like bookmarks. You place them in your document and use them to easily return to *marked* locations in your file.

Note: Some instructions below mention selecting **Source** from the main menu bar. If you are working in the Editor window, you must instead select **Edit** from the Editor main menu bar.

Creating and Naming a Location Mark

To create and name a location mark in your document, do the following:

1. Place the cursor at the position to be marked.
2. Select **Source** from the main menu bar.
3. Select **Name a mark**.
4. Enter the name of the new mark in the text entry field of the **Name a mark** dialog box.

5. Click on **OK**.

Finding a Named Location Mark

To move the cursor to a named mark, do the following:

1. Select **Source** from the main menu bar.
2. Select **Goto** from the resulting pull-down menu.
3. Select **Mark** from the cascading menu.
4. A **Locate - Mark** dialog box appears, highlight the name of the mark you want to move to.
5. Click on **OK**.

Using a Quick Location Mark

To mark one cursor position quickly, you can use the **Quick Mark** feature as follows:

1. Move the cursor to the position where you want to set a mark.
2. Select **Source** from the main menu bar.
3. Select **Set quick mark** from the resulting pull down menu. A quick mark is set at the chosen position in your file.

To find a quick mark, do the following:

1. Select **Source** from the main menu bar.
2. Select **Goto** from the resulting pull down menu in the Source view or **Locate** in the Editor window.
3. Select **Quick mark** from the next pull down menu. The cursor is moved to the position marked by the quick mark.

You can have only one quick mark per document view. Setting a new quick mark will cause a previous quick mark to be lost.

You can also find a quick mark using the procedure described in **Finding a Named Location Mark** above. The quick mark will appear in the list of named marks as **@QUICK**.

Chapter 7. Find Text or Locations in a File

Find a Specific Line in a File

A quick way to find a specific line in a file is to use the **Locate Line** facility.

1. Select **Source** (or **Edit** if working in the Editor window) from the main menu bar.
2. Select **Goto** (or **Locate** if working in the Editor window) from the resulting pull-down menu.
3. Select **Line...** from the cascading menu. The **Locate line** dialog box appears. The **Line number** text entry field displays the line number of the current cursor position.
4. Replace the current line number with with another number, and click on **OK**. The **Locate Line** dialog box disappears and the cursor moves to the beginning of the chosen line.

Note: If you try to go to a line number that is greater than the number of lines in the file, the cursor moves to the beginning of the last line in the file.

Find Text

To search for an item in your document or file view, first do the following to open the Editor - Find and Replace dialog:

- If working in the Source view:
  1. Select **Source** from the main menu.
  2. Select **Find** from the resulting pull-down menu.
  3. Select **Find and Replace** from the cascading menu. The Editor - Find and Replace dialog opens.
- If working in the Editor window:
  1. Select **Edit** from the main menu.
  2. Select **Find and Replace** from the resulting pull-down menu. The Editor - Find and Replace dialog opens.

Specify the item to be searched for by doing the following:

© Copyright IBM Corp. 1998

1. Type a search item in the **Find** text entry area. This can be a word, a partial word, or a sequence of such. You can also enter a pattern you want to match, provided that the pattern follows the rules of Regular Expression.
2. Select your desired options. If you enter a pattern in the **Find** text entry area, you must also select the **Pattern match** check box.
3. Click on **Find**. If the entered text or pattern is found, the cursor moves to either the next or previous occurrence of the search item, according to your chosen search direction.

Find and Replace Text

To search for an item in your document or file view, first do the following to open the Editor - Find and Replace dialog:

- If working in the Source view:
  1. Select **Source** from the main menu.
  2. Select **Find** from the resulting pull-down menu.
  3. Select **Find and Replace** from the cascading menu. The Editor - Find and Replace dialog opens.
- If working in the Editor window:
  1. Select **Edit** from the main menu.
  2. Select **Find and Replace** from the resulting pull-down menu. The Editor - Find and Replace dialog opens.

Specify the item to be found and replaced by doing the following:

1. Type a search item in the **Find** text entry area. This can be a word, a partial word, or a sequence of such. You can also enter a pattern you want to match, provided that the pattern follows the rules of Regular Expression.
2. Type the replacement word in the **Replace with** text entry field.
3. Select your desired options. If you enter a pattern in the **Find** text entry area, you must also select the **Pattern match** check box.
4. Click on **Replace**. If the entered text or pattern is found, the cursor moves to either the next or previous occurrence of the search item, according to your chosen search direction, and replaces the found text according to your selections.

Chapter 8. Manage Files in the Editor

Create a New File

If working in the Source view, you can open a new file by doing the following:

1. Select **Source** from the main menu bar.
2. Select **Open...** from the resulting pull-down menu. The Open File dialog appears.
3. Enter the name of the file in the **File name** entry field. If a file by this name does not yet exist, the editor creates it.
4. Decide how you want your new file displayed. Your choices will usually be:

Open in the Current Pane	The file opens into the current Source view pane.
Open as a Floating Pane	The file opens into a floating Source view pane.
Open as a Workbook Section	The new file is opened into the Source view pane of its own Workbook Section.

5. If you want the new file to be included as part of the current project, select the **Add to Project** check box, along with one of the following options:

Add Source to Configuration
Add Source to Target Directive
Add Source to Source Directive

6. Select the **Open** push button to create the file with your selected options. The new file appears in the chosen Source view.
7. Start entering text into your new file.

If working in the Editor window:

1. Select **Source** from the main menu bar.
2. Select **New...** from the resulting pull-down menu. The New dialog appears.
3. Use the check boxes and selection lists to select the options you want.
4. Click on **New** in this dialog box. A new, empty Editor window appears.
5. You can start entering text for your new file in this view.

Note: Files created in the Editor window may not be known to or recognized by a project defined in the IDE.

© Copyright IBM Corp. 1998

Open an Existing File

If working in the Source view, you can open an existing file by doing the following:

1. Select **Source** from the main menu bar.
2. Select **Open...** from the resulting pull-down menu. The Open File dialog appears.
3. Use the file selection fields to pick the name of an existing file, or enter the file name directly in the **File name** entry field.
4. Decide how you want your file displayed. Your choices will usually be:

Open in the Current Pane	The file opens into the current Source view pane.
Open as a Floating Pane	The file opens into a floating Source view pane.
Open as a Workbook Section	The new file is opened into the Source view of its own Workbook Section.

5. Start editing text in your file.

If working in the Editor window, you can open an existing file by doing any of the following:

- Drag a file icon from a File Manager window onto an Editor icon.
- At a command line prompt, type

  ```
  iedit filename.ext
  ```

  where

  ```
  filename.ext
  ```

  is the name of the file you want to edit.
- If the Editor is already running, do the following:
  1. Select **File** from the Editor menu bar.
  2. Select **Open...** from the resulting pull-down menu.
  3. Use the file and directory selection lists to select the file you want to open.
  4. Click on the **OK** push button to open the selected file for editing. An Editor window opens showing the chosen file.

Note:	The Editor will not try to open a file that is already loaded in the Editor.

Save a File

If working in the Source view, save the file in the current Source view by doing the following:

1. Select **Source** from the main menu bar.
2. Select **Save File** in the resulting pull-down menu.

If working in the Editor window, do the following:

1. To save one file:
   a. If you have more than one Editor window open, select the window that contains the work you want to save.
   b. Select **File** from the Editor menu bar.
   c. Select **Save** from the resulting pull-down menu.
   d. If you are saving an existing file, the file is saved under its current name. If you are saving a new file, the **Save as** dialog box appears. Enter a new name for the file and click **OK**. The new file is saved under this name.
2. To save one file to a different file name:
   a. If you have more than one Editor window open, select the window that contains the work you want to save.
   b. Select **File** from the Editor menu bar.
   c. Select **Save as...** from the resulting pull-down menu. The **Save as** dialog box appears.
   d. Enter a new name for the file and click **OK**. The new file is saved under this name.
3. To save changes to all files you are currently working with, do the following while in any open Editor window:
   a. Select **File** from the Editor menu bar.
   b. Select **Save all** from the resulting pull-down menu. Existing files are saved under their current file names. For each new, un-named file saved, the **Save as** dialog box appears. Enter the new name of the file and click on the **OK** push button to save the new file under that name.

Embed Another File into the Current Document

The Get operation embeds another file into the file currently open in the editor. The **Get**operation will not load any new profiles along with the file being inserted. Profiles already loaded remain in effect and are applied to the file being inserted.

If working in the Source view, embed another file into your document by doing the following:

1. Move the cursor anywhere on the line above where the inserted text is to appear.
2. Select **Source** fromthe main menu bar.
3. Select **Get...** from the resulting pull-down menu. The **Get File** dialog box appears.
4. Select the desired file and click on **OK**. The inserted text appears on the line(s) after the cursor position.

If working in the Editor window, embed another file into your document by doing the following:

1. Move the cursor anywhere on the line above where the inserted text is to appear.
2. Select **File** from the Editor menu bar.
3. Select **Get...** from the resulting pull-down menu. The **Get** window appears.
4. Select the desired file and click on the **OK** push button. The inserted text appears on the line(s) after the cursor position.

Work with Multiple Documents

The Editor lets you have several documents (file views) open for editing at the same time. These views can be of different documents, or different views of the same document. Displaying multiple views of the same document can help you perform functions like cutting and pasting from one section to another. Any changes you make to one file view are automatically updated in all other views of that document. You can also carry out functions like cutting and pasting between views of different documents.

If you are working in the Source view, you can work with and move between file views in any combination of tiled Source view panes, floating Source view panes, and Source views of other Workbook sections. To do so:

- Designate which Source view pane you want to place the file into when you first open that file.
- Use the source selection list found in the top frame of each Source view pane to change to and work with other files already opened for editing in the IDE

If you are working in the Editor window, you can manage different file views with the following facilities:

- The Split window option lets you split the Editor window vertically or horizontally to view multiple file views at the same time. Select **View** from the Editor window menu bar, then **Split** from the resulting pull-down menu.
- The Ring Manager lets you arrange files into groupings, or rings. You can easily move between the rings, or between files within a given ring. To open the Ring Manager window, select **Windows** from the Editor window menu bar, the **Ring manager**from the resulting pull-down menu.

Set Breakpoints in Program Source Files

Breakpoints can only be set on lines that contain executable statements.

To set a breakpoint on a line, do the following:

1. Click on the line on which you want to set the breakpoint.
2. Select **Source** from the main menu bar.
3. Select **Set Breakpoint** from the resulting pull-down menu. A breakpoint is set on the chosen line. If the **Show Breakpoints** option is enabled, the first non-blank character on the line changes to show a red or green background, indicating the presence of a breakpoint.

You can choose to show or hide breakpoint settings in a file. To do so, do the following:

1. Select **Source** from the main menu bar.
2. Select **Hide Breakpoints** from the resulting pull-down menu to hide all breakpoints in the file from view. The breakpoints themselves still exist, but can't be seen.
3. Select **Show Breakpoints** from the resulting pull-down menu to show all breakpoints in the file.

Note: The **Hide Breakpoints** and **Show Breakpoints** options affect only background coloring of the first non-blank character on the lines containing breakpoints. If you have enabled the Annotation Column, breakpoint settings are still shown in the Annotation Column regardless of other options settings.

If you have enabled the Annotation Column, breakpoints can be set by clicking in the annotation area of lines chosen for breakpoints. Annotation Column behavior is as follows:

- The first-time single-click in the annotation area changes the color of the annotation box to red, indicating a breakpoint is set and active.
- Another single-click in the box changes the color of the box to green, indicating that the breakpoint, while set, is temporarily inactive.
- Subsequent single-clicks toggle the box from red to green, activating or deactivating the breakpoint.
- A double-click clears the breakpoint.

Issue Editor Commands

You can use Editor commands to customize the Source view or the Editor window, search for or change text in your document, or perform many other functions. The Editor is fully programmable through the use of an extensive Editor command set. Elaborate sequences of Editor commands are probably best saved to a macro or profile, but you can also issue commands to the Editor without having to write a macro or profile.

To issue an Editor command:

1. Press the **Esc** key to bring up the Editor command line at the bottom of the Source view or the Editor window.
2. Type your Editor command on the command line, then press Enter to perform the command.

To recall a previously-used command:

1. While in the command line, press the up or down arrow key. Previously used commands will appear in the message window.
2. Select the command you want to reuse, then press Enter.

Issue Multiple Editor Commands

The **mult** command lets you issue multiple Editor commands on one command line. To use the **mult** command to issue multiple commands in one step:

1. Change cursor focus to the editor command line.
2. Type the following command:

   ```
   mult ;command_1 ;command_2 ;command_3
   ```

   where command_1 and so on are the commands you want to run.

3. Press Enter.

There are many commands you can use in the editor command line. For a complete listing of Editor commands, see the online help.

Chapter 9. Customize the Editor Appearance and Function

Enable the Annotation Column

To enable the Annotation Column in the Source view, do the following:

1. Select **Source** from the main menu bar**.**
2. Select the **View Settings** from the resulting pull-down menu. A floating Settings pane appears.
3. Expand the **Window Controls** entry, found near the top of the Settings pane.
4. Select **Annotations column**. The Annotations Column appears to the left of each line in the Source view.

Change Editor Tab Settings

You can easily change the Editor tab stops to suit the requirements of the document you are working on.

To change the tab stops, you must first open the **Editor parameters** dialog.

- If you are working in the Source view:
  1. Select the **Workbook** tab section.
  2. Select the **View Settings** bar.
  3. From the left side pane (Workbook, Views with Settings), select **View Source**.
  4. Select **Other Settings** to display selections.
  5. Select **Editor parameters...**. The **Editor parameters** dialog appears.
- If you are working in the Editor window:
  1. Select **Options** from the Editor window main menu bar.
  2. Select **Editor parameters...** from the resulting pull-down menu. The **Editor parameters** dialog appears.

In the **Editor parameters** dialog, do the following:

1. Find and highlight the **Tabs** entry in the parameters selection list.

   The current value for the **Tabs** parameter is displayed below the selection list. For example, a **Tabs** parameter of EVERY 8 indicates that a tab stop occurs at every eighth character position on a line.

© Copyright IBM Corp. 1998

2. Click on **Change...** . The **Set Option Tabs** window appears.
3. Enter your new tab stops into the **Set Option Tabs** window.

   For example, if you want a tab stop at every third position, enter EVERY 3. You can also enter tab stops at irregular intervals.
4. Click on **OK** to save the new **Tabs** parameter setting and close the **Set Option Tabs** window.
5. Click on **OK** to close the **Editor parameters...** window.

Change the Base Editor Font

To customize the base font used by the Editor, you must first open the Font dialog.

- If you are working in the Source view:
  1. Select **Source** from the main menu bar**.**
  2. Select the **View Settings** from the resulting pull-down menu. A floating Settings pane appears.
  3. Expand the **Other Settings** entry, found near the bottom of the Settings pane.
  4. Select **Fonts...**. The **Font** dialog appears.
- If you are working in the Editor window:
  1. Select **Options** from the Editor window main menu bar.
  2. Select **Font...** from the resulting pull-down menu. The **Font** dialog appears.

In the **Font** dialog, do the following:

1. Use the selection lists to choose a font name, style, and size.
2. When you are satisfied with your chosen font, click on **OK**.

   The chosen font is applied to the current editor window and to all future edit sessions. To cancel all changes, click on **Cancel** instead.

Changing Token Display Attributes

The Editor uses different font and character effects to emphasize different parts of a program.

To customize the attributes assigned to different types of document elements, you must first open the Token attributes dialog.

- If you are working in the Source view:
  1. Select **Source** from the main menu bar**.**
  2. Select the **View Settings** from the resulting pull-down menu. A floating Settings pane appears.
  3. Expand the **Other Settings** entry, found near the bottom of the Settings pane.
  4. Select **Token Attributes...**. The Token Attributes dialog appears.
- If you are working in the Editor window:
  1. Select **Options** from the Editor window main menu bar.
  2. Select **Token Attributes...** from the resulting pull-down menu. The Token Attributes dialog appears.

In the **Token Attributes** dialog, do the following:

1. Change token display attributes by doing the following:
   a. Select the token type whose attributes you want to change from the **Token type** selection list.
   b. Select the **Foreground** radio button if you want to change the foreground color, otherwise select the **Background** radio button.
   c. Select the color button with the color combination that you want to apply to the token type. The text in the sample window changes to show the effect of your choice.
   d. Select any combination of character emphasis (reverse, italic, bold, or underline) for the token type. The text in the sample window changes to show the effect of your choice.
2. Repeat the above steps to change the display attributes of other token types as needed.
3. If you want the new display attribute settings saved for use in future editing sessions, enable the **Save settings** check box. Otherwise, the new settings will stay in effect only for the current editing session, and are lost when you close the Editor.
4. When you have finished changing token display attributes, click on **OK**.

Customize the Keyboard

Use can customize the keyboard when working in the Editor by using **set key** or **set action** commands.

A **set key** command, lets you remap the connections between the physical keyboard and the logical keyboard. This remapping is global and affects all Editor files.

You can use **set key** to use the editor with a keyboard that does not have all of the required keys. For example, if your keyboard does not have the F11 and F12 keys, you can remap the Alt+1 and Alt+2 key combinations to have the same effect:

```
set key.A-1 F11
set key.A-2 F12
```

Similarly, you can use various accented characters that are not on the keyboard by assigning them to keys. The following example assigns lowercase and uppercase umlauts to the Alt+A and Ctrl+A key combinations, respectively. The lowercase umlaut is represented by ascii code 132, and the uppercase umlaut is represented by ascii code 142.

```
set key.A-A 132     lowercase umlaut
set key.C-A 142     uppercase umlaut
```

Note: Ctrl and Alt key combinations do not differentiate between uppercase and lowercase characters. For example, using Alt+a or Alt+A in any Editor command yields the same result.

Set action command, lets you program the logical keyboard, and assign a command or set of commands to a key. This programming relates to a single file, so the same key can have a different effect for different files.

When a key is pressed, the Editor performs an action dictated by the corresponding logical key. Many keys have default actions. Any key can be assigned an action using the **set action** command. For example, the following command makes the F7 key scroll the text in a window up one screen:

```
set action.F7 scroll screen up
```

The operating system reserves some keys, such as F1, and F10.

The **set key** command is sometimes more useful than a **set action** because remapping using **set key** is global. For example, you could define the Ctrl+H key as the help key in all files by using the following Editor command:

```
set key.C-H F1
```

Change Text and Background Color Palettes

The Editor uses a set of up to sixteen colors for text and background display. A default color palette is supplied with the Editor, but you can modify this color palette or define and save other color palettes as you require. Changes to the color palette apply to all open Source views and Editor windows.

To use or define a different color palette in the Source view, you must first open the Color Palette dialog.

- If you are working in the Source view:
  1. Select **Source** from the main menu bar**.**
  2. Select the **View Settings** from the resulting pull-down menu. A floating Settings pane appears.
  3. Expand the **Other Settings** entry, found near the bottom of the Settings pane.
  4. Select **Color Palette...**. The Color Palette dialog appears with the name of the current color palette displayed in the **Palettes** field.
- If you are working in the Editor window:
  1. Select **Options** from the Editor window main menu bar.
  2. Select **Color Palette...** from the resulting pull-down menu. The Color Palette dialog appears with the name of the current color palette displayed in the **Palettes** field.

In the Color Palettedialog, do the following:

- To change to another, previously-defined color palette:
  1. Click on the down-arrow button at the right side of the **Palettes** field to see a list of defined palettes.
  2. Select the name of the color palette you want from this list.
- To create a new color palette:
  1. Select the color button that represents the color to be changed.
  2. Change the values in the **Red**, **Green**, and **Blue** color component fields until the color button selected in step **a** shows the color you want. Permissible values are integers from **0** to **255** inclusive, where **0** indicates lowest intensity of color, and 255 indicates highest intensity of color.
  3. Repeat steps **1** and **2** for each color that you want to change.
  4. When you finish defining colors for the new palette, click on the **Add** button. The Add a Palettedialog box appears.
  5. Enter the name of the new color palette in the Add a palettedialog box, and click **OK** to return to the Palettedialog box. The name of the new color palette now appears in the **Palettes**field.

- To delete a color palette:
  1. Click on the down-arrow button at the right side of the **Palettes** field to see a list of defined palettes.
  2. Select the name of the color palette you want to delete from this list. If the deleted color palette is the palette in effect when you first opened the Palettedialog box, the colors in that deleted palette remain in effect until you explicitly select another color palette.

Click on **OK** to close the Palettedialog box and apply your changes to the Editor.

Use Profiles to Modify Editor Behavior

A number of profiles are loaded each time you start the Editor or open a file into it. These profiles contain various commands and parameter settings that control how the Editor looks and behaves.

You can directly modify the default Editor load profiles to your own personal preferences, but your modifications may be lost if you later reinstall the software. To avoid losing your load profile modifications, we recommend that you do the following:

1. Do not directly modify the standard load profiles that come with the Editor.
2. Instead, create your own load profiles using an lxu filename extension according to the following naming conventions:

profinit.lxu	User profile that runs when the Editor is first initiated.
profsys.lxu	User profile that runs each time a new file is loaded into the Editor.
*xxx***.lxu**	User-defined language profile that runs each time a file of type *xxx* is loaded into the Editor. For example, if your personalized load profile is to apply to .cpp files, your personalized load macro will be called cpp.lxu. **Note:** When you load the Editor with file type *xxx*, the Editor loads the *xxx*.lxu user load profile immediately after the *xxx*.lxl standard load profile.

Personalized load profiles can be stored in the **macros** directory, but we recommend that you store them in your own directory.

Note: AIX

Uses the **/usr/vatools/macros** directory.
OS/2

WIN

Uses the *product_installation***macros** directory.

Creating a Personalized Profile
This example shows you how you can create and customize your personal profile in, for example, the **profinit.lxu** file.

Perform the following steps:

1. Create a new file in the Editor with the following lines. The first line of the profile must be a comment, as shown. Do not forget the quotation marks around each command line.

```
/* profinit.lxu */
'set blockdefaulttype rectangle'
'set toolbar.my_button HELP 'Add 5 lines to end of file' 5 mult ;bottom ;add 5'
'set actionbar.~Queries.Class 5 query class'
'set actionbar.~Queries.F~onts\tCtrl+Z 5 query fonts'
'set action.c-Z query fonts'
```

1. Save the file in the **macros** directory with a name of **profinit.lxu** and quit the Editor. Your new Editor customizations take effect the next time you start the Editor.
2. To cancel your customizations, edit the **profinit.lxu** file to remove the unwanted Editor commands, or erase the file completely.

Storing a Personalized Profile
Personalized profiles should be stored in your own directory. This helps ensure that your profiles are not lost if you later reinstall the Editor. It also simplifies profile management when the Editor is used in a network environment.

To store your profiles in your own directory, do the following:

1. Create a directory in which to store your personal profiles, for example:

```
mkdir \my_prof
```

1. Create your personalized profiles and store them in this directory.
2. Modify the AIX LPATH4 or OS/2 WIN **CPPLPATH4** environment variable to include the new directory. If this environment variable does not already exist, create it. For example:

AIX export LPATH=/my_prof:$LPATH4

OS/2 WIN set CPPLPATH4=d:\my_prof;d:*product_install_path**macros;*

1. OS/2 WIN Reboot your system.

Change Editor Load Profiles

You can select the load profiles applied to your file. The Editor can use both system and user profiles to customize Editor functions. In addition, programming language profiles can also be specified so that parsing functions are available to highlight the tokens appropriate to a given programming language.

To change profiles in the Editor proceed as follows:

1. Select the **Workbook** tab section in the Source view (select **Options** from the menu in the Editor window and then proceed to step 4 of this instruction).
2. Select the **View Settings** bar.
3. From the left side pane (Workbook, Views with Settings), select **View Source**.
4. Select **Profiles** to display selections
5. Select **Change Profile**. The Change Profile dialog box appears.,
6. Select any combination of:

`System`	Apply the system load profile (profsys.lxu) to the file.
`User`	Apply the user preferences load profile (profile.lx) to the file
`Language profile`	Apply a language profile to the file. If this check box is selected, a list of available language profiles is enabled. Scroll through the list and select one of the listed profiles.

7. When you are satisfied with your selections, click **OK**. To cancel all changes, click **Cancel**.

Write Macros for the Editor

You may want to write an editor macro to customize aspects of the Source view or the Editor window, and to automate lengthy or frequently used sequences of commands.

Macros are written using a combination of REXX commands and Editor commands and parameters. Macros used by the Editor, along with other example macros, can be found in the **macros** directory. You can use these system macros as a guide when creating your own macros, as follows:

1. Decide on a name for the macro, according to its purpose:
   - If the macro will be run when a specific event occurs, such as when the Source view or the Editor window is started, or when a file is saved or loaded, use the naming conventions described in Macros and Profiles, and Create and Store Personalized Profiles.

     Note:A profile is a special kind of macro.
   - If you are creating an ordinary macro that will be run using the **macro Editor** command, name the macro *filename*.lx, where *filename* can be any name except profile.
2. Create a new text file.
3. Place the commands you want to issue in the text file.

   The first line of the macro must start with a comment. Comments begin with /* and end with the */ symbols. Editor commands within the macro must be enclosed in single or double quotation marks. If you need to extract information from the Editor, use the **extract** command.
4. Save the file.
5. Ensure that the file is in a directory listed in the AIX LPATH4 or OS/2 WIN **CPPLPATH4** environment variable.

Note: Unless you issue a **sshow** command, the Editor does not refresh information displayed in the Source view or the Editor window until after a macro finishes running.

See the **repeats** sample macro for a detailed example of how to create a macro.



Printed in the USA