

DATA SHEET

MICRO FOCUS SERVER EXPRESS™ REMOTE DEVELOPMENT OPTION

Executive Overview

HIGH PRODUCTIVITY DEVELOPMENT FOR LINUX AND UNIX DEVELOPERS

Micro Focus Server Express™ is the platform of choice for creating scalable high-performance business applications on UNIX and Linux. Server Express helps to dramatically reduce development and deployment costs, and provides increased service levels through state-of-the-art components like FaultFinder and Server for COBOL and Server for SOA.

Micro Focus' Server Express for Eclipse has greatly improved the productivity of developers of these systems by providing a functionally-rich, integrated development environment using the popular Eclipse IDE. However productivity is still somewhat limited by the need for remote desktop access to the Linux or UNIX system hosting Server Express or the need to develop locally on a workstation and then deploy to the server for testing purposes.

Micro Focus Server Express Remote Development Option combines a high-performance workstation based development environment with integrated support for the entire edit-compile-debug cycle on the Linux or UNIX server.

Key Benefits

- Fully integrated, COBOL sensitive development environment quickly ramps the productivity and skill sets of COBOL and non-COBOL developers.
- Standard Eclipse functionality allows the re-use of existing Eclipse skills and enables non-COBOL developers to quickly develop and modernize existing COBOL assets.
- Source code stays on the target environment removing time-consuming source control activities.
- Debugging occurs on the target environment making full use of the server-based systems such as databases or middleware while the development environment runs at full speed on the local workstation.
- Lightweight deployment combined with the centralized "update site" reduces the overall cost of ownership.

Detailed Feature Overview

COBOL AND ECLIPSE – ENTERPRISE DEVELOPMENT POWERHOUSE

Eclipse's open source community has made it the development platform of choice for many developers for several years, especially for Java development. Its extensible framework ensures that developers constantly improve and extend the framework to make it work the way they (the development community) want it to – as a result it is a highly efficient development environment.

With Net Express for Eclipse and Server Express for Eclipse, available as add-on options to the base products, COBOL became a first-class citizen in the Eclipse framework providing the key capabilities that COBOL developers demand to streamline their work. This integration has been taken to another level with the Server Express Remote Development Option (RDO) which has a lightweight client-side COBOL development environment where the application remains on the target Linux or UNIX server.

REMOTE DEVELOPMENT SOLUTION

Server Express RDO has a set of lightweight client plug-ins for Eclipse that provides the COBOL sensitive features such as the COBOL perspective, debug client, and build management. This communicates with the Remote Development Server (RDS) which uses the COBOL Server and Server Express functionality to execute builds and provide debug services. Source code remains on the server but accessed by the Eclipse editor via Samba mounted drives.

Remote Development



Eclipse IDE

- Editor with Syntax Colorization & Background Syntax check
- Remote Debug with full COBOL capability
- Initiate builds
- Source stays on server
- Installed from RDO Server



COBOL Server

- Server for COBOL/SOA
- Server Express
- COBOL build using local systems
- Debug Engine

Samba

- Remote source access

Eclipse Remote System Explorer

- Remote server communications
- Eclipse Update Site

Figure 1: Logical Architecture

COBOL PERSPECTIVE

“Perspectives” are Eclipse’s technique for ensuring the developer has access to all the right tools at the right time. By providing a COBOL perspective, the COBOL programmer has direct access to:

- COBOL project types
- COBOL editing, debugging, and building
- Templates that accelerate the entry of code, reducing typing time and errors
- Outline views of the program structure for quick navigation
- COBOL language syntax colorization, Watch windows, breakpoints and more when editing and debugging COBOL applications

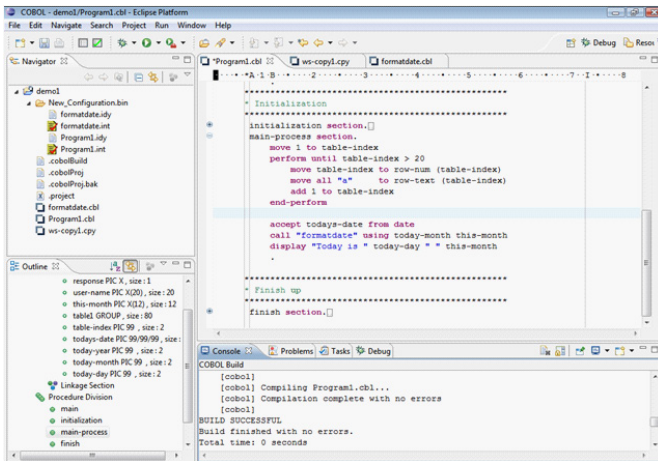


Figure 2: Example COBOL Perspective view

COBOL Editing

SYNTAX HIGHLIGHTING

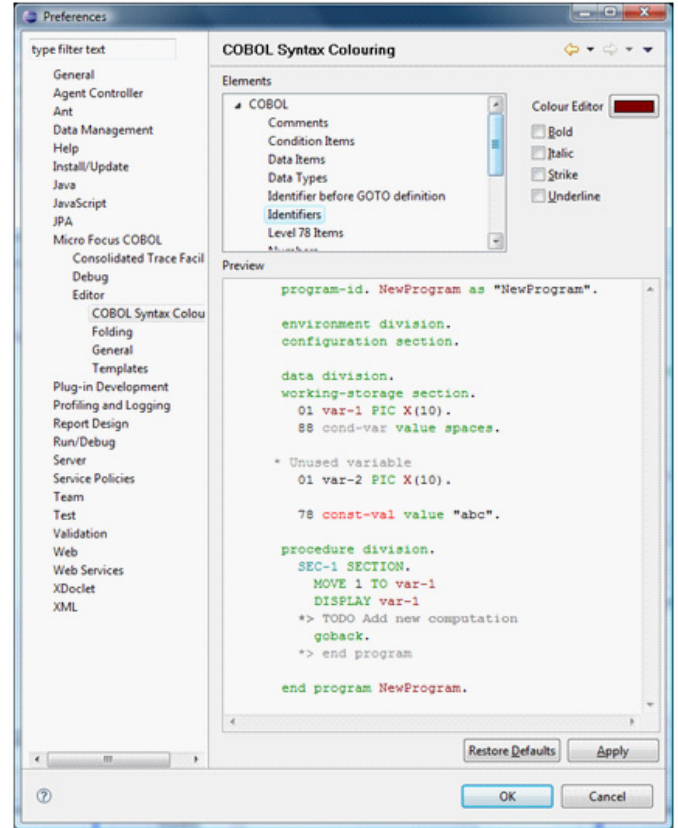


Figure 3: Colorization is sensitive to the compiler COBOL Syntax Highlighting

With the Eclipse plug-in, the editor is fully aware of COBOL syntax and colorizes the text appropriately. Colorization is sensitive the compiler directives specified in the source code and the user can tailor the colorization scheme to suit their preferences.

COBOL OUTLINING

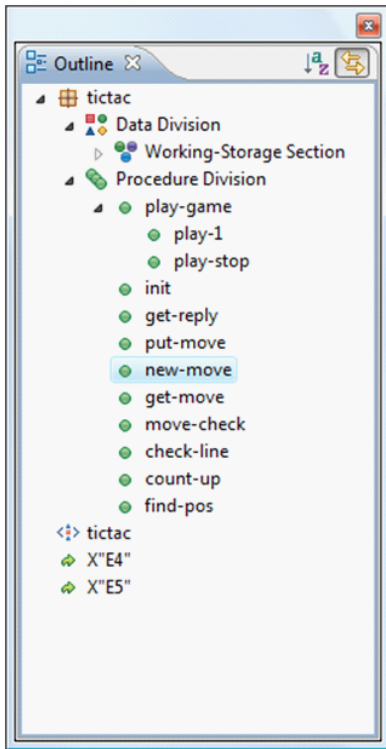


Figure 4: The outline view

The Outline view provides a collapsible tree view of the program showing the various sections of the program, including the data division and procedure division. Clicking on the various items in the outline view provides quick access to that point in the source code.

COBOL breakpoints and watchpoints can be set directly from items in the outline view for optimum programmer efficiency. There is also the ability to start searching for references of COBOL data items from the outline view with the results displayed in the 'Search Results' view exactly as you would see for a standard Java search.

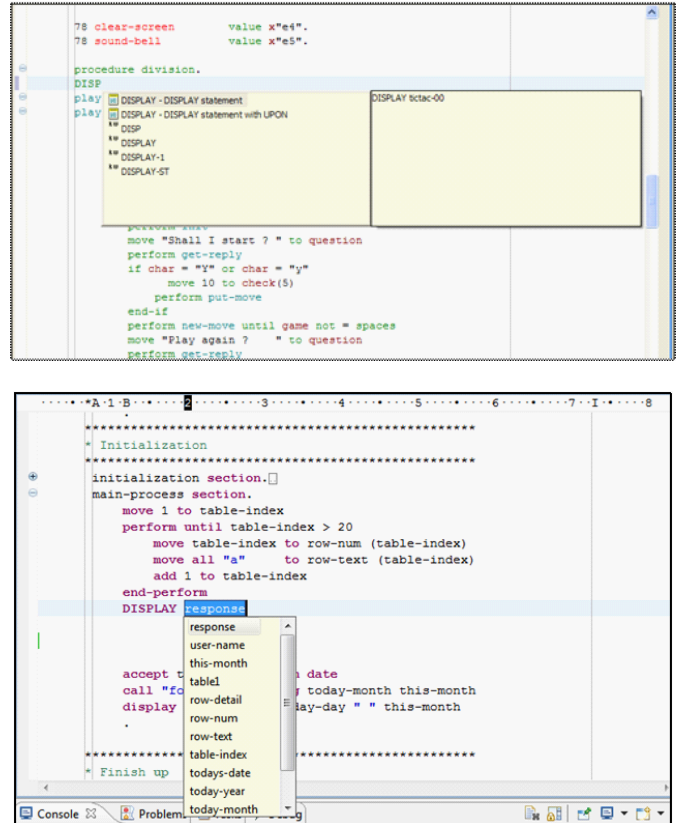


Figure 5: COBOL templates in action

COBOL TEMPLATES

When the developer is entering or modifying COBOL source, pressing a keyboard shortcut provides quick access to standard COBOL statements. These can be inserted directly into the source saving time and reducing errors.

BACKGROUND PARSING

While changes are being made to the application, a background process continually performs a syntax check while the programmer continues working. This is completely transparent – there's no slow-down in the editor – but immediately flags any errors preventing failed project builds occurring later.

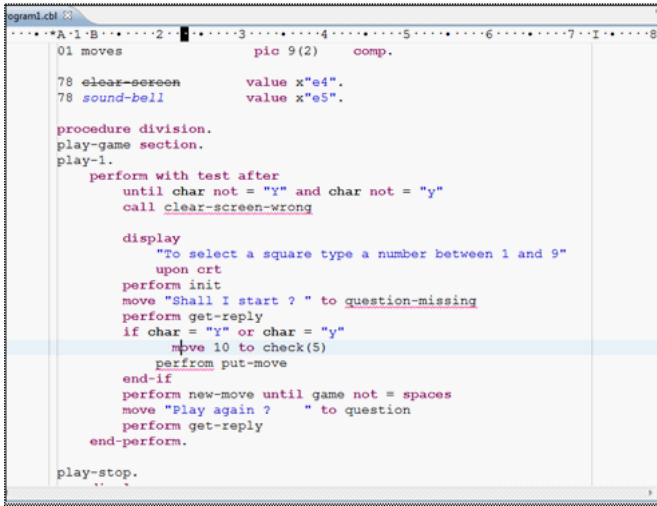


Figure 6: Instant feedback

Any compiler errors found by the background task are reported in the standard Eclipse console.

COBOL HELP

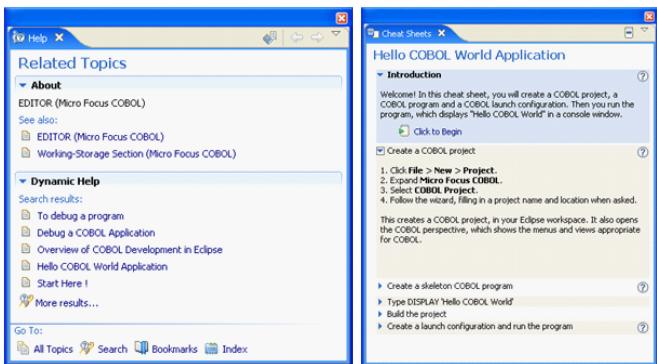


Figure 7: Help on COBOL and Cheat Sheets

Selecting a COBOL keyword and then selecting help presents instant assistance on that COBOL syntax along with links to more general information on creating and debugging COBOL applications. "Cheat sheets" are also available to provide a quick tutorial on the most commonly performed development tasks.

REFERENCE LOOKUPS

With access to the dictionary built by the COBOL compiler, the Eclipse "Find References" feature allows programmers to quickly see where data items are referred to making it easier to spot the impacts of changes being made.

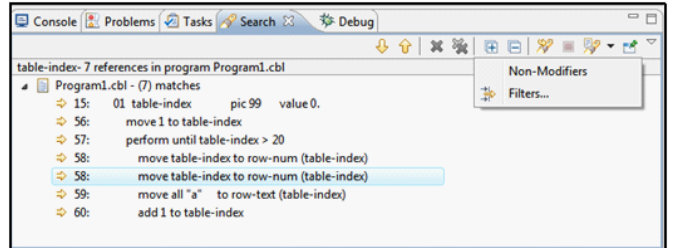


Figure 8: References View

Using filters on the references view allows for quick identification of where data items are used – an important impact-analysis consideration when making changes to code, especially if the programmer is not familiar with every line of code.

DEBUGGING

With full support for debugging COBOL applications, the developer can use Eclipse with Micro Focus COBOL's powerful debugging tools to investigate problems. The application runs on the same server platform as the final production release so ensuring the debug environment behaves as it would when deployed – contrast this with other alternatives that may require debugging on Windows before deployment on Linux which may behave differently. Three options are provided for debugging:

- Starting the application to be debugged on the remote machine from the Eclipse client.
- Wait for attachment from a running process CBL_DEBUG_START
- Attach to a process running on the remote server

Although the application is running remotely, the debug client provides all the rich capabilities expected in Eclipse including viewing the call stack, viewing/changing variables, tooltip data etc. For maximum flexibility, the application could be running in 64bit mode or the server while the client is 32bit.

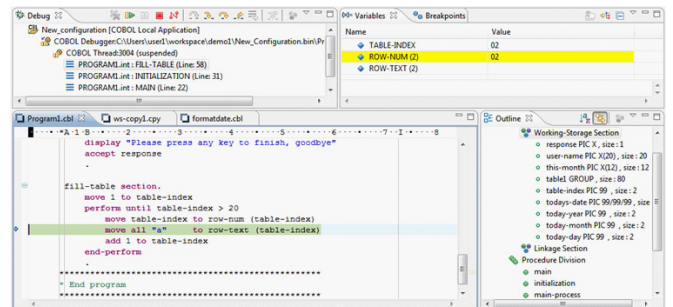


Figure 9: Debugging a COBOL application

```

fill-table section.
move 1 to table-index
perform until table-index > 20
  move table-index to row-sum (table-index)
  move all "a" to row-sum (table-index)
  add 1 to table-index
end-perform
End program
    
```

Figure 10: Viewing the definition of a variable.

Build System

BUILD MANAGEMENT

When building COBOL applications it is sometimes important that build steps are executed in the correct sequence. For example, there may be different versions of the same sub-program in different directories and selecting the one to use for a particular build determines the application behavior. The COBOL Eclipse plug-in makes it easy to manage the directory build order using project preferences.

REMOTE BUILDS

Using the RDO, COBOL projects are built on the remote server. This ensures that the application is built correctly for the target platform and has access to resources on available on that platform. Build progress is communicated back to the controlling IDE and pre- and post- build scripts can be executed on the remote machine.

Platforms

ECLIPSE CLIENT

Microsoft Windows	Windows XP
	Windows Vista
	Windows 7
	Windows Server 2003
	Windows Server 2008 R2

ECLIPSE SERVER

IBM AIX, Sun Solaris, HP-UX (Itanium), Red Hat Linux, SuSE Linux.

PRE-REQUISITES

- Eclipse v3.5 or later (if user already has Eclipse, Eclipse v3.5 included in the package)
- Eclipse client: Sun Java 1.6 or later,
Server: Sun Java 1.6 for non-AIX platforms, IBM Java 1.6 on AIX,
- Samba configured for Windows access to the remote Linux or UNIX server
- Server Express v5.1 WS4 or later (installed on the server)

For additional information please visit: www.microfocus.com