SAPinsider

This article appeared in the Oct ▪ Nov ▪ Dec 2012 issue of *SAPinsider* (http://sapinsider.wispubs.com) and appears here with permission from the publisher, WIS Publishing.

WIS pubs

Turbocharge Your ABAP Development with Innovation from Eclipse

Introducing ABAP Development Tools for SAP NetWeaver

by Karl Kessler, SAP AG

Karl Kessler (karl.kessler@sap.com) joined SAP AG in 1992. He is the Product Manager of the SAP NetWeaver foundation — which includes SAP NetWeaver Application Server, Web Dynpro, the ABAP Workbench, and SAP NetWeaver Developer Studio — and is responsible for all rollout activities. Currently, Karl's work centers on how SAP NetWeaver powers SAP ERP and SAP Business Suite software, with a focus on lifecycle management.

The Eclipse development framework offers a range of benefits for Java application development. And with SAP NetWeaver Developer Studio, SAP customers have been able to take advantage of Eclipse's flexibility and extensibility in their own Java applications. Eclipse also serves as the basis for the latest innovations in SAP development environments, including the SAP HANA studio and SAP NetWeaver Cloud. But what about ABAP development, which makes up a significant portion of application development in SAP customer environments?

With SAP NetWeaver 7.31, support package stack 4, SAP introduces ABAP development tools for SAP NetWeaver, also known as "ABAP in Eclipse." ABAP in Eclipse brings Eclipse capabilities to ABAP Workbench (SE80) functionality. This moves ABAP, which is ripe for productivity and usability improvements, into a complementary alignment with SAP's other Eclipse-based development environments.

Here we'll take a look at the basic development environment setup requirements for ABAP in Eclipse, and delve into some key features so you can hit the ground running with Eclipse in your own ABAP development projects.

Setting Up an "ABAP in Eclipse" Development Environment

The first step toward using ABAP in Eclipse is to set up a personal development environment. This might seem a bit unusual for SAP developers who are accustomed to predelivered or bundled development environments. ABAP in Eclipse, however, requires that you take the additional step of installing Eclipse on your desktop or laptop, separately from the SAP product. ABAP in Eclipse supports the Eclipse release train, meaning that you can use any Eclipse version from 3.6 to 3.8. The Eclipse website (**www.eclipse.org/indigo**) hosts the different versions that are available, the latest of which (as of this writing) is Eclipse Indigo.

Once you've installed Eclipse, you can download the ABAP in Eclipse tools, which are available from SAP Service Marketplace (**http://service.sap.com/swdc**),[1] and install them using the installation wizard in Eclipse. ABAP in Eclipse also requires some additional Eclipse plugins, which are documented in the installation guide available in SAP Note 1718399 — RTC of ABAP development tools for SAP NetWeaver, and can be downloaded from the Eclipse update site (**http://download.eclipse.org/releases/indigo**) and installed with the Eclipse installation wizard.

[1] Use the search term, "ABAP Development Tools for SAP NetWeaver 1.0."

Note: ABAP in Eclipse currently requires a 32-bit runtime environment. (You can, however, run this environment on your 64-bit operating system and hardware.)

Enhanced Browsing for Viewing Development Objects

After setting up your development environment, you can begin your development activities. With ABAP in Eclipse, development begins with an ABAP project. The ABAP Workbench, in contrast, requires no project concept, since all the ABAP artifacts are contained in ABAP packages.

When you create a project in ABAP in Eclipse, you establish a connection to an existing ABAP back end to gain access to the development objects (such as programs, function modules, and classes) in the ABAP packages there. All the destinations from the SAP logon dialog are displayed as available connections.

Once you have established a connection, you can use the Eclipse Project Explorer (see **Figure 1**) to browse and view the contents of the back-end ABAP packages, such as your local objects (package $TMP, for example), and add them to your project.

You can then select a development object contained in an ABAP package and display it in the tool appropriate to the activity you want to perform (for instance, the source code editor if you want to modify the code). The outline view, which is located beneath the project explorer view to simplify navigation, displays the structure of the selected development object (such as a list of the program's variables) and enables you to navigate directly to that element in the source code.

The ABAP in Eclipse environment also enables you to work with several ABAP projects in parallel, which means you can work with multiple back-end repositories simultaneously — each

Eclipse serves as the basis for the latest innovations in SAP development environments, including the SAP HANA studio and SAP NetWeaver Cloud.

The Origins of Eclipse-Based Development in SAP Environments

Initially, SAP used Eclipse as a tooling platform to support its portfolio of Eclipse plugins for developing SAP NetWeaver 04 Java-based applications. The original intent was to provide an end-to-end application development experience that could follow both the Java Enterprise Edition standard and the SAP program model based on frameworks such as Web Dynpro on the front end, Web services as the communication and integration layer, and remote function calls (RFCs) on the SAP back end.

In subsequent SAP NetWeaver releases, SAP continued to leverage the flexibility and extensibility of Eclipse. SAP extended its basic program model by creating a central repository and build services, and enabled SAP NetWeaver Developer Studio, which hosts all of the Eclipse plugins from SAP, to provide direct design-time support for SAP NetWeaver Business Process Management (SAP NetWeaver BPM) and SAP NetWeaver Process Integration (SAP NetWeaver PI). Combined, these enhancements allowed developers to carry out every development activity in the Java context in a homogeneous development environment.

With Eclipse firmly embedded in Java development in SAP environments and serving as the basis for new SAP development environments, extending Eclipse-based development to the ABAP world was a natural next step. Early prototypes, featuring code editing, browsable ABAP packages and classes, and debugging support, received an enthusiastic reception at SAP TechEd Demo Jam sessions over the years. However, a comprehensive, deeply embedded integrated development environment for ABAP already existed — the ABAP workbench (transaction SE80), which runs inside the SAP GUI and provides immediate access to any repository object, whether it is a simple ABAP program or a complex Web Dynpro artifact.

So SAP used Eclipse to address ABAP workbench limitations related to SAP GUI, and to provide a bridge between Eclipse-based development task and SAP GUI-based development tasks. This approach preserves the rich functionality of the ABAP workbench, while leveraging the innovative UI capabilities of Eclipse for more efficient programming.

project opens in a separate tabstrip within the object browser, as shown in Figure 1. In contrast, with the classical ABAP Workbench, each back end would require a separate SAP GUI session.

Improved Functionality for Managing Your Source Code

Source editing with ABAP in Eclipse is simple, straightforward, and intuitive. The ABAP in Eclipse source editor integrates many of the features of the ABAP editor, including syntax coloring and syntax outlining, into an Eclipse UI. Once you have browsed to an ABAP program or function module in the project explorer view and opened the object in the source editor, you can easily and efficiently modify it using familiar ABAP functionality such as code completion and code templates (see **Figure 2**).

ABAP in Eclipse improves coding efficiency through display and navigation features. Eclipse's multi-document interface enables you to open several ABAP development objects in parallel. You can also use this multi-document capability to freely arrange a variety of layouts in the object browser to enable easy source code comparisons. In addition, you can use the outline view or the project explorer view to quickly navigate to the corresponding definition of an object element in the source code, saving you from having to search or scroll through the code.

ABAP in Eclipse uses display improvements to enhance the popular "where-used list" feature from the classical ABAP Workbench, which allows you to select a particular variable that is used in one or several programs and generate a list of source locations where the variable is used. Additionally, ABAP in Eclipse provides a more refined way to display the search results, allowing you to drill down into the corresponding programs and add source line numbers, for instance (see **Figure 3** on the next page).

Streamlined Testing and Debugging

Once you have completed your development work, testing and debugging your application couldn't be simpler. The ABAP debugger is completely integrated into ABAP in Eclipse. You simply add a few breakpoints in the ABAP source code by using the context menu in the editor or by double-clicking the beginning of a source line. Then you execute your code by

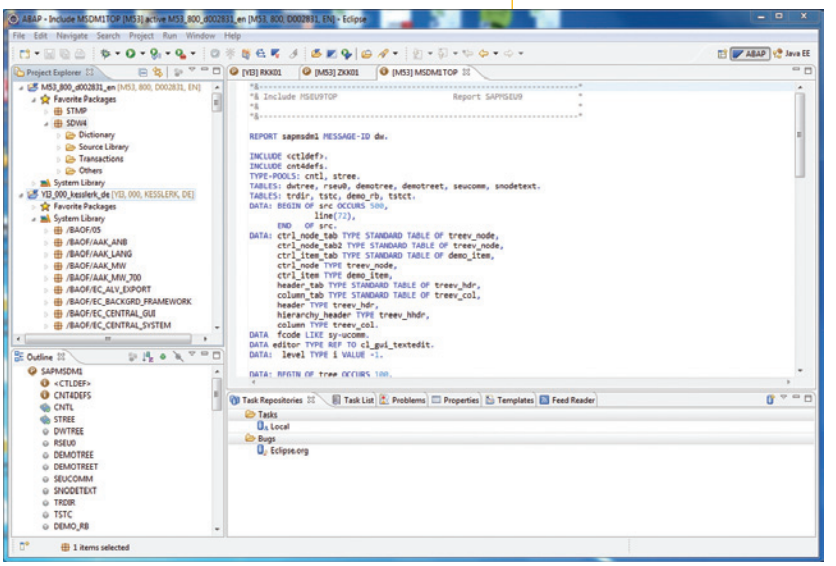

FIGURE 1 ▲ Viewing ABAP development objects in Eclipse

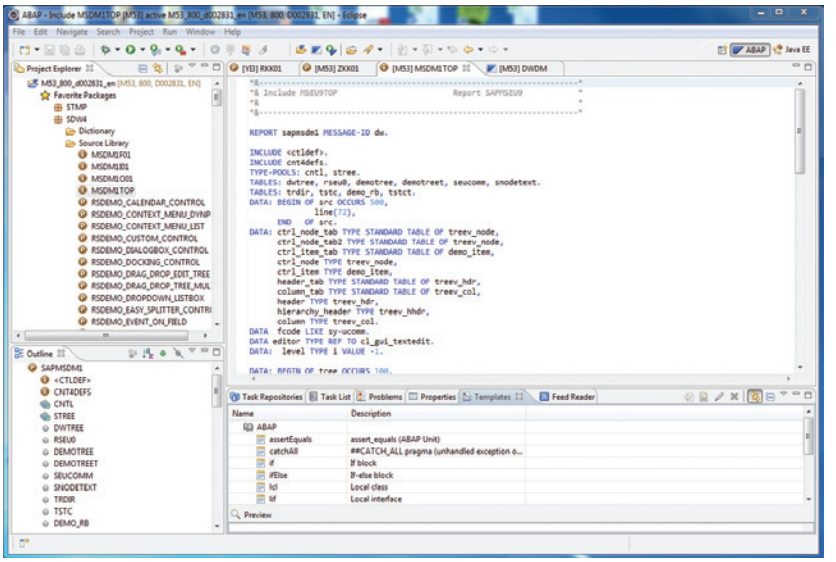

FIGURE 2 ▲ Using ABAP templates in the Eclipse source editor

Tip: Searching for the development object you want to execute is simple with the type-ahead search feature of ABAP in Eclipse. You just launch the search dialog, type a significant prefix, and the system immediately displays a hit list in an Ajax-style interaction. Since many RFC function modules start with the RFC prefix, browsing the repository becomes really easy.

selecting the development object from the project explorer view and selecting "Execute" from the context menu. Alternatively you can use the "Run" button which is widely used in the Java environment for similar purposes. ABAP in Eclipse responds by launching the ABAP runtime environment within an SAP GUI session embedded in the Eclipse environment, and executing the code.

Once the ABAP runtime environment hits a breakpoint, execution stops and the ABAP debugger then opens (see **Figure 4**). The ABAP debugger in Eclipse is a true two-process debugger that cleanly separates the debugger and debuggee sessions. It offers all the capabilities you would expect from the classical ABAP debugger, including a source code display with the program counter that points to the statement being executed, the call stack, a variable display, and a list of breakpoints. In addition, typical ABAP features, such as a display for monitoring internal tables, are fully supported, and traditional debugger instructions such as "SINGLE STEP," "STEP OVER," and "RETURN" (from subroutine) are available using standard Eclipse icons.

Note: ABAP in Eclipse also includes unit testing functionality to support an agile approach to ABAP development.

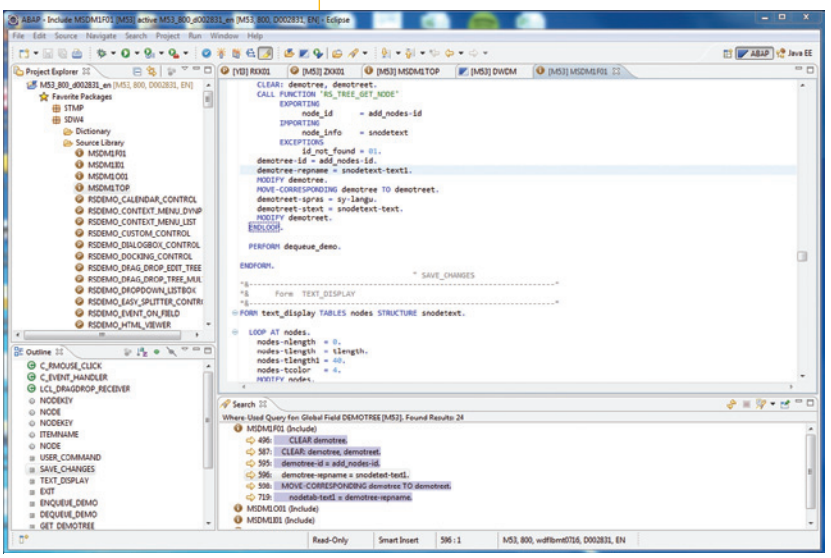

FIGURE 3 ▲ Displaying the where-used list results in Eclipse

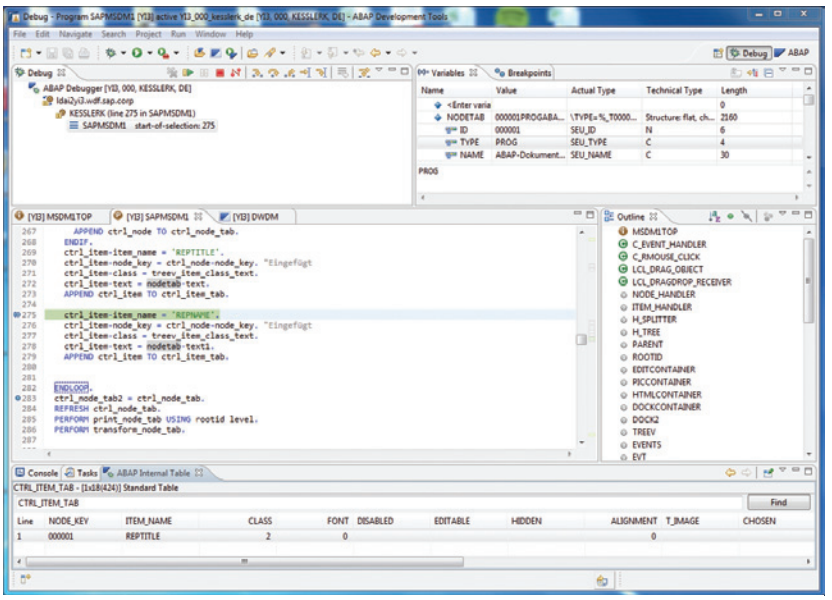

FIGURE 4 ▲ The ABAP debugger in Eclipse

Next Steps

ABAP in Eclipse takes the best elements of the classic ABAP Workbench functionality and the Eclipse UI and combines them to create a more powerful toolset than what was previously available. For example, ABAP in Eclipse includes extension points, allowing partners and customers to build their own plugins using the ABAP development tools software development kit — a capability that was not available with the ABAP Workbench.

Going forward, SAP will continue to add functionality to ABAP in Eclipse. For example, while not all ABAP development artifacts are currently supported, SAP is working hard to fill the gaps.[2] SAP also plans to add support in subsequent ABAP in Eclipse releases for additional popular tools frameworks, such as Web Dynpro. In addition, SAP will use discussions on SAP Community Network, as well as feedback from various customer engagements, to help determine implementation directions that best support its customers.

Ready to get started? Download a trial version from SAP Community Network at **http://scn.sap.com/docs/DOC-29607.**

[2] Note that ABAP in Eclipse displays development objects in an embedded SAP GUI session to enable Eclipse-based navigation even for unsupported artifacts.