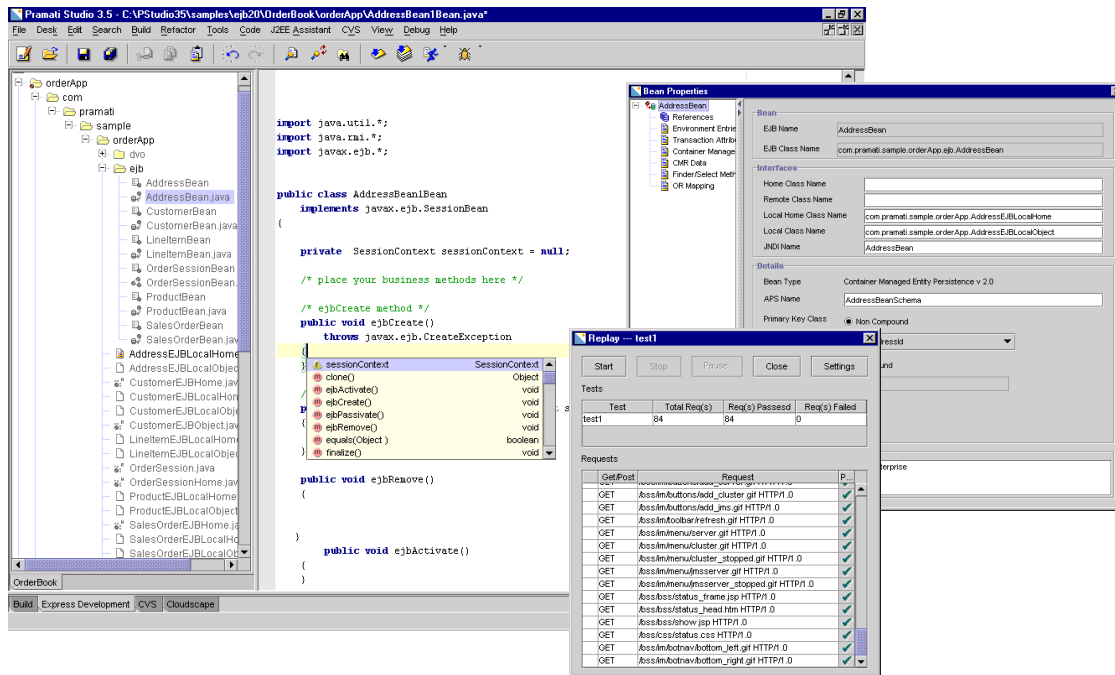




Pramati Studio is a pure Java server-side IDE to create, test, debug, package and deploy applications on the Java™2, Enterprise Edition application server platform. The new Studio 3.5 is designed for J2EE-centric power development, with J2EE refactoring aids that preserve J2EE semantics, functionality and unit testing tools, Application Analyzer, enhanced Debugger, and a new built-in Pramati Server 3.5.



Key Features

- Create, package, deploy, test and debug J2EE applications on an integrated full-featured J2EE 1.3-compatible application server.
- Iterate through J2EE 1.3 development life cycle, with Express Development™-driven generation and updation of archives.
- Advanced code formatting, Pretty Printing, Online Validation and Aggressive Editing feature.
- J2EE refactoring for managing component-level changes and cross-component impact.
- Extensive base of Java coding aids for rapid coding and error-free code generation.
- Rapid web component development with JSP coding aids.
- HTML and XML validation and tag correction.
- Integrated J2EE semantic validation and correction.
- JSP test tool record and playback app testing with JSPTestTool
- Cactus™ and JUnit™ integration

- Full-feature J2EE debugger which lets you debug the application on local as well as remote VMs including features as conditional breakpoints, watch points
- J2EE Application Analyzer
- UML support through ArgoUML™ and Rose™ import facility.
- Deploy tool that enables a single code base for seamless deployment on multiple application
- Wizards to generate EJB 2.0 beans including beans with container managed relationships and message driven beans.
- Migration Tool that converts project sources and archives from non-standard platforms to pure J2EE structure.
- Packaging Tool that auto-generates all mandated EJB 2.0 (and 1.1) packaging XMLs for deployment on any J2EE platform.
- Version control from inside IDE with CVS and VSS integration.
- Comprehensive view of the application through Bean Properties.
- DB Navigator to connect to and manage database tables.
- Published Studio Framework and Tool APIs to plug in own tools.

System Requirements

Any platform that supports JDK 1.3.1_01. Min 128 MB RAM (256 MB recommended). Min 50 MB hard disk space (application size not included).

Platform	JDK version
WinNT 4.0, SP5	JDK 1.3.1_01, 1.3.1_02 and 1.4
WinNT 2000	JDK 1.3.1_01, 1.3.1_02
Linux RedHat 6.2	JDK 1.3.1_01, 1.3.1_02
Solaris 7, 8	JDK 1.3.1_01, 1.3.1_02

JDBC drivers for other databases may be provided by respective vendors.

Database	JDBC Driver
Oracle 8i	Thin driver V 8.1.5.0.0, MERANT DataDirect Connect JDBC V2.2
Oracle 9i	Oracle thin driver 9.0.1.1.0
Informix 7.3.0 TC3	Driver V 2.10 JC1N361
Cloudscape 4.0	Cloudscape 4.0 RMI JDBC Driver
MS SQL Server 7	MERANT DataDirect Connect JDBC V2.2, JTurbo Driver 2.3

J2EE Centric Power Editing

- Online validation and error highlighting of code for violation of J2EE specifications. Assistance in form of pop-up suggestions and automatic error fixing.
- Context sensitive code completion with QuiCode (a.k.a. *code insight* or *code pop-up*).
- Context-sensitive help for Java constructs through hyperlinked Javadoc and tags.
- Online semantic validation and correction (checks for Java and JSP semantics, HTML compliance, XML DTD/schemas).
- Formatting and indentation of sources. Multiple options to customize formatting to the standards of a particular project.
- Browse classes, variables, methods, and URL with Source Code Browsing that has special J2EE-centric approach.
- Auto indexing of source files.
- Abbreviations for common constructs such as "SOP()" for "system.out.println()".
- Block commenting and uncommenting, import completions, extend existing classes and implement existing interfaces.
- Color-coding support for all J2EE files.

J2EE refactoring of components and impact check

- J2EE aware refactoring in addition to Java refactoring
- Preserves J2EE semantics and updates all related XMLs.
- Refactoring manages loose coupling through meta-data in XMLs.
- Hides details associated with how Java constructs are named.

The following usages in Studio can be refactored in a J2EE context:

- Rename package, class, methods and fields.
- Change method signature.
- Extract superclasses and interfaces.
- Move anonymous class to inner.

Integrated J2EE semantic validation and correction

- Online validation based on J2EE 1.4, EJB1.1, and EJB 2.0 specifications. Any deviation from the bean provider contract given in EJB specification marked as error in Editor.
- Write custom platform 'services'
- Live validation for spec-mandated naming and semantics
- Validation of integrity of components and meta-data

Java coding aids

- Online validation of compilation errors based on Java version 1.4 specification with optional force validation.
- Code completion of various types of code fragments as classnames, methods and variable names.
- Customize Java code formatting rules as preferences in the IDE to provide standard look & feel.
- Extensive source-code browsing of standard Java sources.
- Mark and surround code block with standard expressions as if/else and try/catch. Studio checks for errors in the block before accepting code change.
- Faster access to methods and fields using method and declaration browsing. Incremental search of field, method and class names through the list.

- Override methods from a superclass and implement methods from an interface.
- Complete JavaDoc support.

JSP coding aids

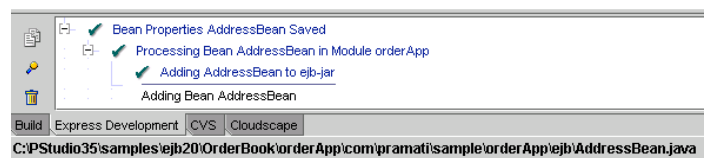
- Syntax and semantic validation and resolution of compilation errors.
- Java code completion. HTML and JSP tag completion options appear when code completion is invoked after typing the JSP start token. Standard Java code completions provided in scriptlets and declaration blocks of a JSP.
- Source code browsing in Java code, custom tags, and linked resources.

XML coding aids

- Wide range of dynamic pages, multimedia presentations, scripting and active content included XML online validation.
- Tag, attribute and value completion.

HTML coding aids

- Online validation of HTML 4.0 compliant configurable errors. Common errors covered are writing HTML in SCRIPT element, ampersands in URLs, incorrect nesting of elements, using a NAME attribute with IMG or FORM, and using lowercase for DOCTYPE.
- HTML tag, attribute, and value completion facility. Empty tags marked.
- Formatting to set the code layout.



Express Development

- Auto-generation and compilation of interfaces.
- Auto-assembly and archive updation of component even as application changes.
- On changes of business logic in class, all related components (JARs, WARs, EARs, XMLs) updated on single click.

EJB, JSP, Servlet and Filter templates

- Create customized, ready-to-code EJB, JSP, servlet and filter templates.
- Add copyrights, information text, custom code, change tracking and logging information to template.

Enterprise Bean generation

- Template-based development of EJB 1.1 and EJB 2.0 beans, including message driven beans.
- Auto-generate session and entity bean skeletons with CMP/BMP for both EJB 1.1 and EJB 2.0.
- Generates Java sources (bean skeleton, PK classes and home, remote, local home, local interfaces). Only business methods to be written.
- Supports session synchronization, re-entrant beans and multiple EJB names.

- Supports all container managed relationships, fields in EJB 2.0.
- Define and publish business methods on appropriate interfaces.
- Define CMRs for all types of cardinality (one/many), directionality (uni/bi) as well as various return types.

Web Services Development tool

Used to quickly publish a Session Bean as a Web service. The Web services runtime used in Apache Axis is bundled with a standard Enterprise Archive that can be deployed on any J2EE Server.

Developing on Multiple Application Servers

Work with multiple application servers and insure past investments in J2EE by maintaining a common code base, based on pure J2EE code with no proprietary extensions. Configure WebLogic™ 6.1, WebLogic 7.0, Oracle™ 9iAS, WebSphere™ 4.0, WebSphere 5.0 and Apache Tomcat™ 4.1 in Studio. Enter the deployment descriptors only once through a common and intuitive UI for all servers. Descriptors are synchronized as bean sources change.

Test-deploy EJB 1.1 and EJB 2.0 components using in-built containers. Seamlessly test deploy components on other application servers.

Test-deploy EJB 1.1 and EJB 2.0 components using in-built containers. Seamlessly test deploy components on other application servers.

Interface Wizard

The Interface Wizard generates all interfaces for EJBs, appropriately exposing selected methods, and automatically updates them to include new business methods.

User Manager

- Build JAAS-based security models into applications.
- Map user-group formations to realm roles during deployment.
- Define users-groups transparently on all configured app servers.

Deploy Tool

Factor out vendor-specific deployment descriptors and processes for all configured application servers. The tool reads target server environment properties and resolves references of EJBs at deploy time. Enables smart Object-Relational (O-R) mapping using DB constraints for EJB 2.0 relationships.

- Complete O-R mapping and other deployment information.
- Auto-resolution of O-R mapping for relationships by inspecting constraints in DB
- Manage unfinished tasks with an intelligent task manager that monitors list of incomplete tasks.

Package Tool

Create standard, portable packaged archives that can be deployed on any application server. Package Tool hides underlying processes that construct the XMLs. Modified sub-components are reflected in archives with a single click.

Query Designer

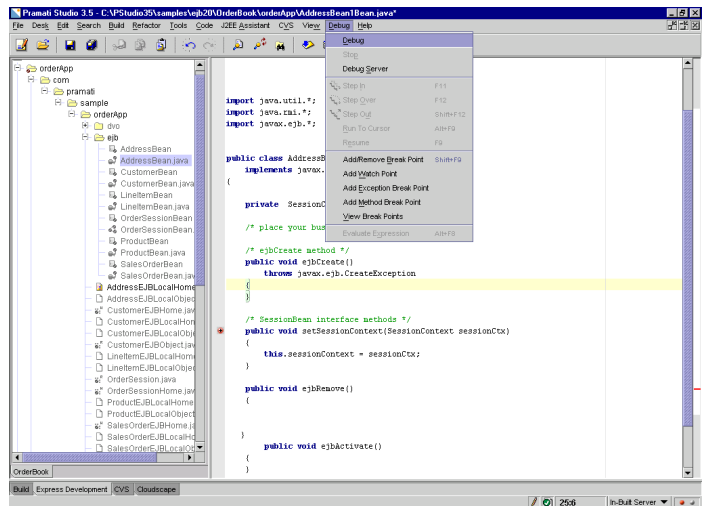
Wizard to create queries based on database schema for EJB 1.1 applications, with support for EJB QL in EJB 2.0 applications. All queries transparently embedded in a generated packaging XML.

- Create EJB QL queries using the Query Designer for entity beans.
- SQL queries converted to application server-specific formats (for example, WLQL™) for transparent migration of applications across servers.
- Specify queries in EJB QL for EJB 2.0 CMP beans.
- An easy to use interface to define and manage multiple types of application resources like database, messages, URLs, Connectors and Mail

Resource Tool

An easy-to-use interface to define and manage multiple types of application resources like database, messages, URLs, Connectors and mail. Define:

- Database resource and connection pools in terms of JNDI name, JDBC driver, URL, user name, and password.
- XA data resources that allow two-phase commit in transactions.
- JMS resource like queues, topics and connection factories to use asynchronous messaging through a JMS Server.
- J2EE Connector API (JCA) Connection Factories to connect to non-J2EE applications using the Resource Adapter Archives (RAR) available on the Server
- Mail resource for using the JavaMail API.



Full-featured J2EE Application Debugger

- Remote debugging of JSP pages and EJB.
- No system code shown.
- Step in from JSP into the EJB method.
- Debug JSP pages at the script level while Servlets execute.
- Set exception and method breakpoints.
- Preserve pre-defined debug configuration sets.
- Enhanced Evaluate Expressions and Variables Watch facilities.
- Drill down variables as the code executes
- Set conditional breakpoints.

J2EE Application Analyzer

- Web-based diagnostics tool runs on built-in Pramati Server.
- Generate diagnostics for analyzing execution patch.
- Drill down execution paths to spot method-level bottlenecks.

Migration Tool

- Support any J2EE server on same code base.
- Migration Tool to move raw sources or packaged archives to other platforms transparently.
- Migrate deployable archives from other servers with package and deployment XML descriptors.
- Migration Readiness Report to identify and eliminate all server-specific lock-ins, for migrating to pure Java platforms.
- Convert source hierarchy into Studio Desk modules with package XMLs and generate deployment descriptors.

Built-in J2EE and JMS Server

- Built-in J2EE 1.3 Server for testing Web and EJB components.
- JMS 1.0.2 compliant Pramati Message Server for testing asynchronous messaging objects.
- Each workstation is fully equipped for complete life cycle development with no additional investment.

Database Navigator

- Add-on tool to view and modify/add tables in any database resource from Studio.
- Table inspection and modification including table characteristics, relationships and constraints.
- Run SQL queries from the DB Navigator performing all routine jobs on the database.
- Cloudscape database is supplied with Studio.

Version Control System Support

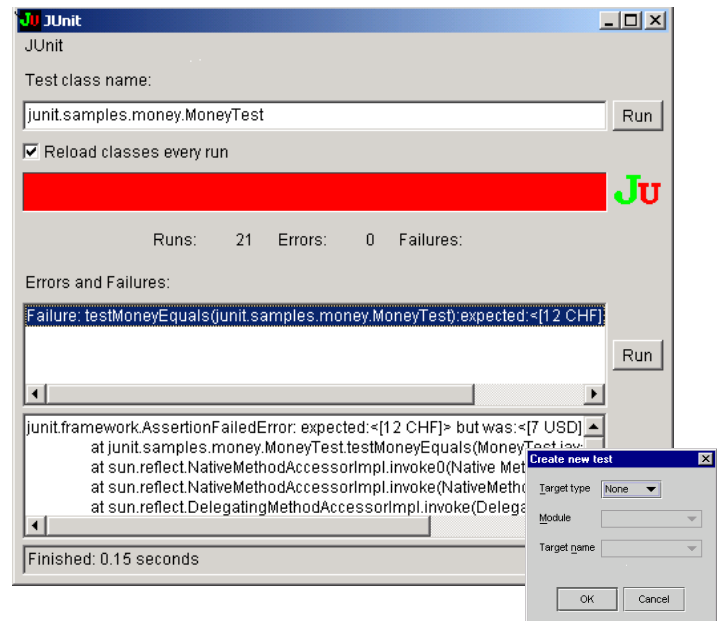
- Plug-in any version control system through published API.
- Out-of-box integration with CVS and Visual SourceSafe.
- Native diff tool with color coding and smart diff in case of reformatting changes.
- Modified files highlighted along with package hierarchy.
- CVS command line within IDE.

Testing Tools

- JUnit for generating JUnit test cases for Java classes. Configure generated test cases to run with deployment or independently.
- JSP Test Tool with simple record-and-playback abstraction for functionality testing of JSP, testing HTML and web server operations.
- Build elaborate test suites and execute with single click.
- Jakarta Cactus included for server-side Java code testing.

UML support through ArgoUML and Rose import

- Create structures to maintain, update the sources while importing XML
- Convert application data to XML version 2.0
- Convert view details to XML



Pluggable Tools

- J2ME development tool-kit with mobile device emulator. All coding aids extended to developing on this platform.
- Reporting tools, such as Crystal Reports and Oracle Reports, integrated with built-in Pramati Server.
- Javadocs generation for applications written in Studio.
- Tools API for power users who want to make their custom tools part of the IDE framework for regular use.

Contact

For further information, visit <http://www.pramati.com>