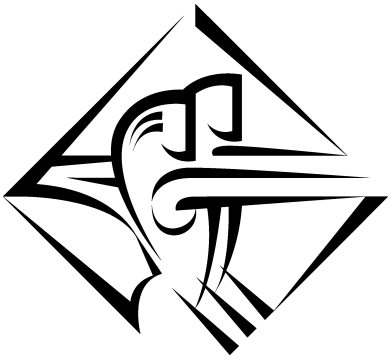
# Tango Enterprise 3



## *User's Guide*

Third Edition
September, 1998

Pervasive Software
12365 Riata Trace Parkway, Building II
Austin, Texas 78727 USA

Telephone: (512) 231-6000
Toll Free: 1 800 287-4383
Fax: (512) 231-6010

Email: info@pervasive.com
Web: http://www.pervasive.com/

PERVASIVE SOFTWARE

**Pervasive Software Inc.**

**Tango(TM) License Agreement**

**IMPORTANT: DO NOT INSTALL THE ENCLOSED SOFTWARE UNTIL YOU HAVE READ THIS LICENSE AGREEMENT. BY INSTALLING THE SOFTWARE, OR AUTHORIZING ANY OTHER PERSON TO DO SO, YOU, AND SUCH OTHER PERSON, IF APPLICABLE ACCEPT THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENSE AGREEMENT, RETURN THE ENTIRE PACKAGE WITHIN TEN DAYS OF RECEIPT.**

1 DEFINITIONS.

a. "Pervasive" means Pervasive Software Inc., a Delaware corporation, 12365 Riata Trace Parkway, Building II, Austin, Texas (U.S.A.) 78727.

b. "You" or "Your Company" means the person or business entity which is licensing the Software pursuant to this Agreement.

c. "Software" means all of the software You or Your Company have received from Pervasive with this License.

d. "Documentation" means the manuals and any other printed material provided by Pervasive with the Software.

e. "License" means the license purchased and granted pursuant to this Agreement.

f. "Effective Date" means the date on which the Software was licensed by You or Your Company.

2 LICENSE AND PROTECTION.

2.1 License Grant. Pervasive grants to You or Your Company, subject to the following terms and conditions, a limited nonexclusive, nontransferable, revocable right to use the Software solely for Your internal development and Your internal testing purposes only on a single-user, desk-top device. You are prohibited from deploying any application programs to other parties that You develop using the Software.

2.2 Documentation. Documentation may not be copied. Pervasive reserves all rights not expressly granted to You or Your Company.

2.3 Protection of Software. You and Your Company agree to take all reasonable steps to protect the Software and Documentation from unauthorized copying or use. The Software source code represents and embodies trade secrets of Pervasive and/or its licensors. The source code and embodied trade secrets are not licensed to You or Your Company and any modification or addition thereto, or deletion therefrom is strictly prohibited. You and Your Company agree not to disassemble, decompile, or otherwise reverse engineer the Software in order to discover the source code and/or the trade secrets contained in the source code.

3 COPIES. You may make a single copy of the Software for archival purposes only. All proprietary rights notices must be faithfully reproduced and included on all copies. You may not copy the Documentation.

4 OWNERSHIP. Ownership of, and title to, the Software and Documentation (including any copies) shall be vested solely in Pervasive. Copies are provided to You or Your Company only to allow You or Your Company to exercise Your or Your Company's rights under this Agreement. Only the License is purchased by You or Your Company, as applicable.

5 RESTRICTIONS. Except as expressly authorized in this Agreement, You and Your Company agree not to use, rent, lease, sublicese, distribute, transfer, copy, reproduce, display, modify, create derivative works of, time share or dispose of the Software or Documentation or any part thereof. You or Your Company, as applicable, may use the Software and Documentation solely for your internal business purposes.

6 ASSIGNMENT. You or Your Companies may not assign or otherwise transfer in whole or in part or in any manner any rights, obligations, or any interest in or under this Agreement without Pervasive Software's prior written consent and any attempted assignment will be void. A merger or other acquisition by a third party will be treated as an assignment.Pervasive may at any time and without Your or Your Company's consent assign all or a portion of its rights and duties under this Agreement to a company or companies wholly owning, owned by, or in common ownership with Pervasive.

7 TERM, TERMINATION. This Agreement is effective from the date You or Your Company open the software envelope and will remain in force until terminated. You may terminate this License at any time by destroying the Documentation and the Software together with all copies and adaptations. This Agreement shall also automatically terminate if You or Your Company breach any of the terms or conditions of this Agreement. You and Your Company agree to destroy the original and all copies of the Software and Documentation, or to return them to Pervasive upon termination of this License. Sections 2.2, 4 and 5 of this Agreement shall survive any termination hereof.

8 LIMITED WARRANTY AND LIMITED LIABILITY.

8.1 Magnetic Media and Documentation. Pervasive warrants that if the magnetic media or Documentation are in a damaged or physically defective condition at the time that the License is purchased and if they are returned to Pervasive within 90 days of purchase, then Pervasive will provide You or Your Company with replacements at no charge.

8.2 Disclaimer of Warranty. PERVASIVE LICENSES THE SOFTWARE PRODUCT TO YOU OR YOUR COMPANY UNDER THIS AGREEMENT SOLELY ON AN "AS IS" BASIS. PERVASIVE MAKES NO OTHER REPRESENTATIONS, CONDITIONS OR WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, REGARDING THE SOFTWARE; PERVASIVE EXPRESSLY STATES AND YOU OR YOUR COMPANY ACKNOWLEDGES THAT PERVASIVE DOES NOT MAKE ANY REPRESENTATIONS, CONDITIONS OR WARRANTIES, INCLUDING, FOR EXAMPLE, WITH RESPECT TO MERCHANTABILITY, TITLE, OR FITNESS FOR ANY PARTICULAR PURPOSE OR ARISING FROM COURSE OF DEALING OR USAGE OF TRADE, AMONG OTHERS.

9 GENERAL CONDITIONS.

9.1 Governing Law. This License Agreement will be governed by, and interpreted in accordance with, the laws of the State of Texas (U.S.A.) exclusive of its choice of law provisions. This Agreement expressly "excludes the United Nations Convention on Contracts for the International Sale of Goods."

9.2 Complete Understanding. This License Agreement sets forth the entire understanding and agreement between You or Your Company and Pervasive and may be amended only in writing signed by both parties. NO VENDOR, DISTRIBUTOR, DEALER, RETAILER, SALES PERSON OR OTHER PERSON IS AUTHORIZED TO MODIFY THIS AGREEMENT OR TO MAKE ANY WARRANTY, REPRESENTATION OR PROMISE WHICH IS DIFFERENT THAN, OR IN ADDITION TO, THIS AGREEMENT ABOUT THE SOFTWARE.

9.3 Waiver. No waiver of any right under this Agreement shall be effective unless in writing, signed by a duly authorized representative of Pervasive. No waiver of any past or present right arising from any breach or failure to perform shall be deemed to be a waiver of any future right arising under this Agreement.

9.4 Severability. If any provision in this Agreement is held invalid or unenforceable, that provision shall be construed, limited, modified or, if necessary, severed, to the extent necessary, to eliminate its invalidity or unenforceablility, and the other provisions of this Agreement shall remain unaffected.

9.5 Export Controls. None of the Software or underlying information or technology may be downloaded or otherwise exported or reexported (i) into (or to a national or resident of) Cuba, Iraq, Libya, North Korea, Iran, or any other country to which the U.S. has embargoed goods; or (ii) to any person or entity on the U.S. Treasury Department's list of Specially Designated Nationals, or the U.S. Commerce Department's Table of Denial Orders, or the

U.S. Commerce Department's Entity List of Missile, Nuclear, and Chemical and Biological Weapons Proliferators, or the U.S. Department of State's Foreign Terrorist Organization List. You agree to the foregoing and you represent the warrant that you are not located in, under the control of, or a national or resident of such country or on any such list. The Software may also be subject to U.S. laws and export regulations of the U.S. government that require an explicit export license prior to any export or reexport of the Software. You agree to obtain any such explicit export license that may be required.

9.6 U.S Government End Users. The Software is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire the Software with only those rights set forth herein. Contractor / Manufacturer is Pervasive Software Inc., 12365 Riata Trace Parkway, Building II, Austin, Texas (U.S.A.) 78727.

9.7 Consequential Damages. WITHOUT LIMITING THE FOREGOING,IN NO EVENT WILL PERVASIVE BE LIABLE TO YOU OR YOUR COMPANY FOR INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO THIS LICENSE AGREEMENT, WHETHER UNDER THEORY OF WARRANTY, TORT, PRODUCTS LIABILITY OR OTHERWISE.

9.8 High Risk Activities. The Software is not fault-tolerant and is not designed, manufactured or intended for use or resale as on-line control equipment in hazardous environments regarding fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of the Software could lead directly to death, personal injury, or severe physical or environmental damage ("High Risk Activities"). Pervasive and its suppliers specifically disclaim any express or implied warranty of fitness for High Risk Activities.

9.9 English will be the controlling language of this Agreement.

Microsoft Software Disclaimer

NO OTHER WARRANTIES, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, MICROSOFT AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH REGARD TO THE SOFTWARE PRODUCT, AND ANY ACCOMPANYING HARDWARE. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE / JURISDICTION TO STATE / JURISDICITION.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL MICROSOFT OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WIHTOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT, EVEN IF MICROSOFT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

**Apple Software Disclaimer**

APPLE COMPUTER, INC. ("APPLE") MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATIONS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A SPECIFIC PURPOSE, REGARDING THE APPLE SOFTWARE. APPLE DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE USE OF THE APPLE SOFTWARE IN TERMS OF ITS CORRECTNESS, ACCURACY, RELIABILITY, CURRENTNESS OR OTHERWISE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE APPLE SOFTWARE IS ASSUMED BY YOU. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME STATES. THE ABOVE EXLUSION MAY NOT APPLY TO YOU.

IN NO EVENT WILL APPLE, ITS DIRECTORS, OFFICERS, EMPLOYEES OR AGENTS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTIONS, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE APPLE SOFTWARE EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMJAGES. BECAUSE SOME STATES DO NOT ALLOW THE EXLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU. Apple's liability to you for actual damages from any cause whatsoever, and regardless of the form of the action (whether in contract, tort [including negligence], product liability or otherwise), will be limited to $50.

**Copyright Notices**

# *Table of Contents*

CHAPTER ONE

# *Introducing Tango Enterprise*

1

*About Tango Enterprise and How It Works*

Tango™ Enterprise is a powerful and easy-to-use tool for creating dynamic, intelligent Web sites that are integrated with popular database systems. With builders, you build solutions by using Tango's intuitive point-and-click, drag-and-drop interface. You can create simple applications in minutes—without ever writing any code. You can customize your application files by adding your own HTML, database queries, and control flow.

This chapter helps you understand Tango and its components. It includes:

- understanding Tango
- Tango Editor
- Tango Application Server
- Tango CGI and plug-ins
- what's new in Tango 3.0
- using Tango documentation
- Tango at a glance
- where to go next.

# Understanding Tango

Tango Enterprise is the industry-leading, application server development environment for generating dynamic Web pages and accessing databases.

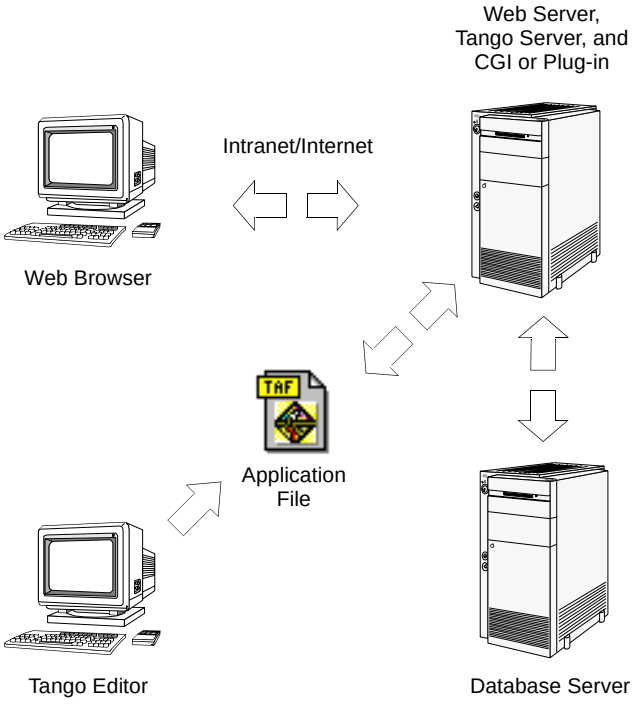
## Tango Components

Tango consists of two main programs: Tango Editor and Tango Application Server, hereafter known simply as Tango Server.

- Tango Editor is the development environment, featuring a complete graphical user interface in which to develop application files.
- Tango Server is an application server which executes application files created with Tango Editor. It works in conjunction with an HTTP (Web) server to return HTML to a Web browser.

For definitions of terms used throughout the documentation, see Appendix A.

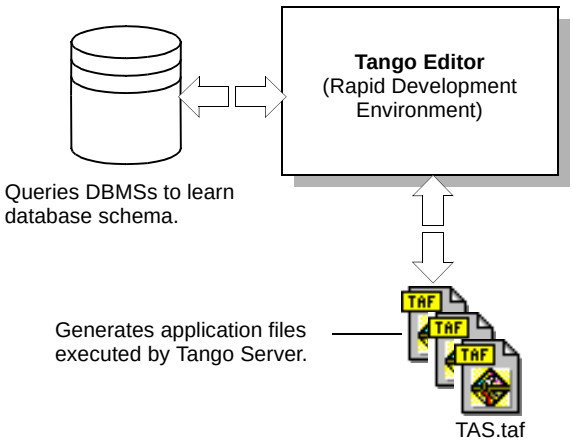
Other components of Tango include the Common Gateway Interface application, or CGI, and Web server plug-ins. The following diagram shows how Tango works with a Web Server.



Web pages viewed in a Web browser can contain forms or links that point to Tango application files created with Tango Editor. When the user submits a form or clicks a link, the Web server receives the request and passes it to Tango CGI or one of the plug-ins that, in turn, sends it to Tango Server. Tango Server then executes the application file, which could involve interaction with a database server. When execution is complete, the results (in the form of HTML contained in the application file) are returned through the CGI or plug-in to the Web server, and then to the user.

## Tango Editor

The Tango Editor development environment provides a point-and-click, drag-and-drop interface in which application files are created for use specifically in Tango Server.

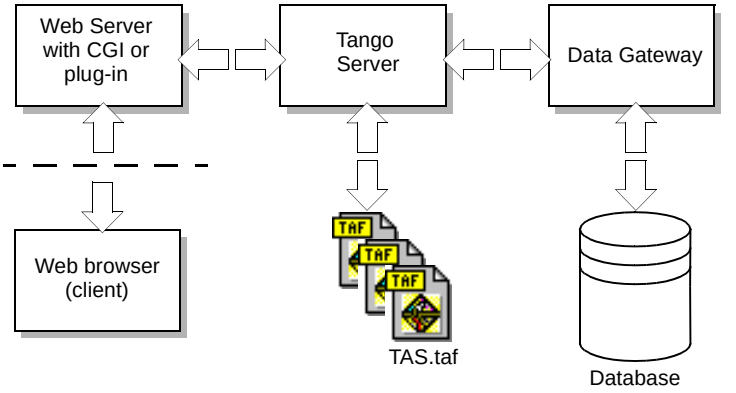



Tango Editor works by querying the database schema and noting the tables and columns of the database. Columns can be dragged into specific actions within the application, thus defining the way in which the database is accessed. No knowledge of SQL or database specifics is required. A series of action results are merged to create the HTML page sent back to the Web server and on to the Web browser.

The distinguishing factor between Tango Editor and other RAD (rapid application development) tools is the fact that Tango Editor is easy to use. Tango Editor contains builders that allow for rapid development not only of the database access and processing but also automatic generation of all the necessary HTML.

## Tango Server

Tango Server works with a Web server to allow for the dynamic creation of HTML based on information contained in databases. Tango Server works as a middle layer between the Web server and browser, and the databases, and includes support for DBMS, e-mail, and other external actions.




Web servers are built to support HTTP requests for HTML pages and other file types that are processed by the Web browser. When Tango Server requests are made to the Web server, they are passed on for processing to the Tango Server via the Tango CGI program or the Web server plug-in.

### *Tracing a Web Request*

To understand these components better, look at the interaction among the components.

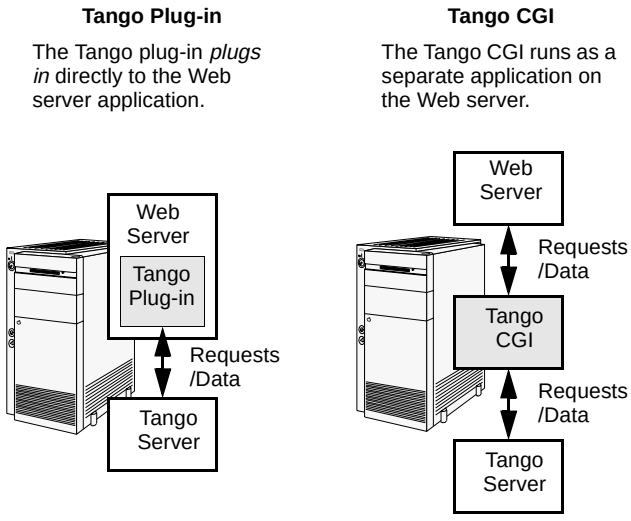
1. A Web browser request is made (using the HTTP protocol). URLs (Uniform Resource Locators) for Tango indicate the location of a Tango application file and optional arguments representing the appropriate entry point and parameters to the application file.
2. A Web server receives that request and recognizes it as a Tango application request (this is normally done through recognition of a suffix to the Tango application file name, but it may be defined differently for specific Web servers).
3. The Web server passes the request to Tango Server through either the Common Gateway Interface (CGI) standard or through a plug-in created to one of two supported standards (NSAPI and ISAPI).

4. Tango Server processes the request and executes one or more application files (previously created with Tango Editor). These files contain one or more actions necessary to fulfill the user request.
5. Databases are accessed via ODBC, or a direct connection is made to an Oracle or DB2 database.
6. Results from a database action are processed by Tango Server using the Results HTML associated with the action (most often, HTML containing placeholders for the pertinent columns). Tango Server performs column substitutions and data manipulation for each row and generates the final HTML for presentation.
7. The composed HTML accumulated by several actions is consolidated into an HTML page that is then returned to the Web server.
8. The Web server returns this HTML to the originating Web browser.

## Tango CGI and Web Server Plug-ins

You can use either the Tango CGI or a Tango Server plug-in to communicate between your Web server and Tango Server. The CGI and plug-in perform the same function: they both communicate a user's request from the Web server to Tango Server, and return the

results to the Web server. However, they differ in *how* they communicate with the Web server software.




The *Tango CGI* (`t3cgi.exe` under Windows, or `t3.cgi` under UNIX) runs as a separate program on your Web server. The *Tango plug-ins*, as the name implies, *plug in* directly to your Web server under Windows NT. One advantage of using plug-ins is speed: data passes back and forth between Tango and the Web server much faster using a plug-in than using the CGI, thereby processing users' requests more quickly.

Two Tango plug-ins are available. The Netscape API (NSAPI) plug-in, `t3ns.dll`, is used with Netscape Web servers, and the Microsoft Internet Information Server API (ISAPI) plug-in, `t3iis.dll`, is used with Microsoft's Internet Information Server (IIS).

## Key Tango Concepts

Five key concepts enable Tango to lead the market in being an easy to use RAD tool. They include:

- visual development environment
- extensibility
- portability
- scalability
- action-based metaphor.

### Visual Tool

Tango Editor provides you with a comprehensible visual development environment that allows you to create application files rapidly. Tango Editor provides builders with which to create applications that find, insert, update, and delete database records. At this level of the development environment, you do not have to write any SQL or HTML to create applications; Tango Editor generates everything for you.

Tango Editor presents application files in an icon-based format. The flow of actions in the file is easily seen and all actions are displayed as distinct icons. This allows you to isolate and alter certain components of the application files you have created while letting the rest stay as they are.

### Extensibility

Tango can be extended beyond its ability to handle predefined types of actions by including different kinds of external actions, including *File*, *Mail* and the general *External* action. This means a Tango application file can interact with external programs, for example, sending e-mail, reading from or writing to a file, launching another application from the command line, or launching programs written in Java. Moreover, results from most external actions are integrated back into Tango in the same manner as database queries.

The External action supports common, platform-specific mechanisms for interacting with other programs:

| Platform | Mechanism |
|---|---|
| Macintosh | **Apple Events** are used to communicate to another process running on the same machine as Tango Server. |
| Windows | A **DLL** (dynamic linked library) mechanism is supported that allows for the creation of Windows 32-bit DLLs using such development environments as Visual Basic and C++. |
| Windows and UNIX | A standard shell, command line interface is provided that allows for **Perl script** and **shell script** integration, along with any standard process that uses StdIn/StdOut for communications. |
| All | **Java** externals are supported in a cross-platform manner allowing for the integration of Tango applications with Java classes and JavaBeans. |

## Portability

Tango provides Web developers with the ability to create true cross-platform Web solutions. Currently, Tango is available for the Apple Mac OS, Microsoft Windows NT, Microsoft Windows 95, AIX (Apple and IBM), SGI IRIX, and Sun Solaris. You can create Tango applications on one environment and deploy them on another: for example, creating Tango application files using Tango Editor on a Windows 95 machine and deploying the application files on a Solaris Web server.

Tango application files are in a binary file format and can be easily transferred from one operating system to another. No recompiling or regeneration of the application files is necessary for execution or editing. This enables you to develop application files on your platform of choice and deploy onto virtually any Web server.

Because of Tango Editor's advanced abstraction of database actions, changing the designated DBMS is a matter of changing the data source associated with the Tango application file.

(The use of Direct DBMS actions—where SQL is put directly into the application file—is generally not portable amongst DBMS systems; however, it is rare for this portion of a solution to have a serious impact on a solution's portability. Tango Search, Insert, Update, Delete, and Transaction actions eliminate the need for Direct DBMS actions in most solutions.)

## Scalability

Tango has the ability to scale to support enterprise solutions. Tango can scale in two areas: Tango Editor allows you to develop enterprise solutions, and Tango Server is able to scale to handle ever-increasing loads.

Tango Editor allows simple solutions to be scaled to handle the complexities of interaction among multiple database systems.

Tango Server provides scalability through its architecture and implementation. Tango Server is designed to maintain high performance under heavy loads. All platforms supported with Tango Server utilize the operating system capabilities for process multi-threading to take advantage of the scaling effects of multiple CPUs. You can deploy multiple application servers and you can deploy Tango Servers, or components thereof, on different host machines. These machines do not have to be identical in configuration or operating system, allowing you to utilize all available resources.

Furthermore, Tango Server architecture utilizes Pervasive's data gateway technologies that allow for deployment across multiple heterogeneous platforms (supporting databases not easily reached from the platform the application server is operating on). This allows for true blocking features for database systems regardless of the capabilities of the DBMS, thus not impeding the performance and scalability of Tango Server. Database driver issues and native database access are achieved with the goal of platform independence and less need for third-party database drivers and middleware.

## Action-Based Metaphor

Tango uses a methodology based on *actions,* which abstracts away from the details of the implementation.

Tango-generated HTML pages are created from the actions in an application file that build the HTML page dynamically based on customer input and results retrieved from a database.

The action-based approach has many benefits:

- **Ease of solution deployment**. You can include multiple HTML segments inside one application file, including HTML forms. This application file is much easier to work with than working with the individual pieces of HTML: it is more easily archived and can be moved for deployment.

- **Shared common actions**. A single action works on one data source and may have attached attributes, which are represented by icons. These attributes are results of the action processing (Results HTML), a page generated if there are no results (No Results HTML), and errors generated by the DBMS (Error HTML).
- **Visualization of control flow**. Database actions are represented by icons and all actions are named. This makes it easy to visualize the control flow taken through the execution of an application. Control actions (such as If, ElseIf, and Loop) allow you to group, loop, and nest actions.
- **Simplification and integration with multiple data sources**. You can tell what different database systems are integrated because they are listed in a clear, iconic format.
- **Grouping of all events associated with performing a single database interaction**. The abstraction of the database action from its database language (normally SQL) allows Tango to deal with desktop database systems that do not support a standard relational language (FileMaker Pro, for example).

# What's New in This Version of Tango

Tango 3 incorporates many new features not present in earlier Tango versions. Here is a list of some important features in Tango 3.

- **Visual Development**. The Tango Editor interface has been revamped to present application files in an easy-to-use and logical fashion.
- **Java Support in External Action**. You can now call Java class files and JavaBeans to be executed on the Tango Server.
- **JavaScript**. You can now execute JavaScript on the Tango Server, which means you can extend the functionality of Tango using this industry-standard scripting language.
- **Transaction Processing**. There is full support for transaction processing within the Tango environment. Transaction processing treats a series of database actions as a single unit.

  For example, if you wanted to transfer funds from a savings account to a checking account, you can think of this as two database transactions. First, you must withdraw from your savings account. Then you must deposit into your checking account. If the update on your checking account failed, then the update on your saving account must be *rolled back*. However, if both actions are successful, you then *commit* the transaction. With Tango, you can build this sort of functionality within your application.
- **Find and Replace**. Tango Editor provides powerful find and replace functionality in application files and supports the use of regular expressions.
- **HTML Enhancements**. Tango has enhanced the non-modal HTML editing capabilities.
- **Data Source Timeouts**. You can specify how long data sources are cached for reuse. Tango automatically closes the data source connection after the specified time period.
- **Variable Scope**. In addition to user variables, Tango now supports additional scopes for variables: local, cookie, domain, and system.
- **Arrays.** Tango has a new variable type that has a row and column structure. This type of variable can be used in many different ways, including storing the results of an action. Returning the value of an array in Results HTML creates an HTML table by default.

- **New Meta Tags**. Many new meta tags have been added. Examples include meta tags that support arrays (including the ability to sort and filter; perform intersections, unions, and distinct processing; and add and delete discrete rows), math and date functions; and the ability to incorporate information from the Web pages that URLs point to.
- **Mail Action**. You can now send e-mail messages directly from Tango.
- **Control Flow Actions**. You can create conditional action executions with If, Else, and ElseIf actions. You can jump from one action to another with the Branch action. You can create loops that execute actions repeatedly while a specific condition is met or for a specified number of iterations. You can group actions together and manipulate them as a unit.
- **Executing SQL Queries from Tango Editor**. You can query a database directly in Tango Editor.
- **Timed Application File Execution**. Tango Server can execute specified URLs at specific times, as well as on shutdown and on startup. The URLs that Tango Server executes could, for example, process application files and start database servers.
- **HTML Syntax Coloring**. You can display all HTML in Tango with different colors, depending on the category of HTML. For example, all meta tags are one color, all standard HTML tags are a different color, and the content is yet a different color. All colors are customizable.
- **Projects**. Projects can contain sub-folders, allowing developers to better organize and manage their solutions. Projects can also include non-Tango files (such as text files and HTML files). This is useful if you use Tango's `<@INCLUDE>` tag to reference HTML stored outside of the application file.
- **Joins Interface in the Search Builder**. Database joins can be separately specified for the Results List page and the Detail page. In earlier Tango versions, these pages shared join criteria by default. This makes the Search Builder more flexible and extends its usefulness.
- **Keyboard Shortcuts**. Numerous keyboard shortcuts are included in Tango Editor, allowing power Tango developers to program even more rapidly than they already do.
- **NSAPI Plug-in for Solaris**. Tango includes an NSAPI plug-in for Netscape servers running on Solaris.

- **Source Code Control**. Tango includes source control commands to check in and check out application files with all the major source code control systems on the market today. Previous versions of Tango required you to use the source control system's client software to perform these basic functions. Including source control menu commands greatly improves Tango's team programming functionality.

# Using Tango Manuals

All Tango documentation, including this *User's Guide*, assumes you are familiar with the basics of using a computer, such as clicking and dragging, and opening and saving files. If you need help with these tasks, you should refer to the documentation that came with your computer.

The Tango documentation does not cover the following topics:

- setting up and configuring your Web server software and your database software
- creating databases and modifying a database structure
- the basics of HTML.

## Getting Started Guide

The *Getting Started Guide* helps you get up and running with Tango as quickly as possible. It covers preparing to install, installing, and setting up Tango. It also covers the various ways to reach Pervasive for product information and technical support.

## User's Guide

The *User's Guide* introduces you to Tango and tells you how to perform the tasks necessary to create your applications. It is your main source of Tango information. Topics covered include Tango basics, including using application files, data sources, snippets, and action builders. Other more advanced topics cover meta tags, variables, specific actions, and using Tango Server. Appendices include a glossary of terms and how to use Java with Tango.

## Meta Tags and Configuration Variables

A companion book to the *User's Guide*, the *Meta Tags and Configuration Variables* manual is an alphabetical listing and description of meta tags and configuration variables.

## Tutorial

The *Tutorials* book helps you understand the key functions of Tango Editor. Using a series of brief lessons that build upon each other, you can progress quickly to learn Tango. The tutorials are designed for users who are familiar with the basics of Windows and HTML, and the operation of a Web server.

## On-line Help

On-line help is available in HTML format from within Tango. It is your most immediate source of information. To access on-line help, do one of the following:

- Choose **Help Home Page** from the **Help** menu.
- Choose **Help** from the context-sensitive menu.
- Open your Web browser, navigate to the Tango Help folder, and choose the file `home.htm`.

# Tango at a Glance

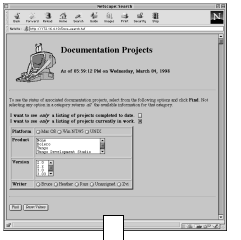
The Internet is a *stateless* network environment. Web pages are not connected to the next, previous, or any other Web page, and the Web browser / client does not maintain a physical connection with the Web page server. Using a combination of HTML and Tango meta tags, Tango provides a means of passing information from one Web page to another.

**BROWSER/CLIENT**

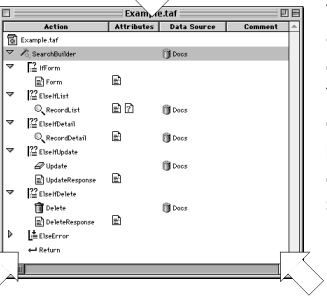
End user values are passed from one HTML page to another in an HTML form (button) or by being appended to a hyperlink (anchor reference).

A form field value, or post argument, is passed in an HTML form with METHOD=POST.

A URL argument, or search argument, is passed in a URL.




**TANGO APPLICATION FILE**

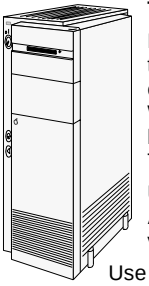



To reference post and search arguments in Tango, use:

**<@SEARCHARG** *argument_name***>** for search arguments only

**<@POSTARG** *argument_name***>** for post arguments only

**<@ARG** *argument_name***>** for both search and post arguments.

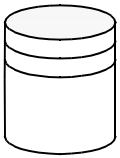


**TANGO SERVER VARIABLES**

End user values and results from the database only last for one execution of an application file or Web page. To keep values for as long as you want, assign them to Tango variables.

Use **<@ASSIGN>** or the **Assign action** to assign values.


Use **<@VAR** *variable_name***>** to reference variables.



**DATABASE**

To capture values from the database, use:

**<@COLUMN** *table_name.column_name***>** in the Results HTML attribute of the database action producing the results, or **<@COL** *column_number***>** anywhere in an application file.

**<@ACTIONRESULT** *action_name column_number***>** anywhere in an application file (only returns the first record of database results).

**<@VAR NAME=resultSet[*x,y*]>** anywhere after a database action and returns an array with all the database results.

# Where To Go Next

If you have not already done so, install Tango by following the instructions in the *Getting Started Guide*.

Then learn the basics of creating and editing application files. Work through the lessons in the *Tutorials* book. With the knowledge you gain, you will soon be ready to begin creating your own Tango solutions.

Refer to this *User's Guide* for all the details about Tango Editor and Tango Server. The *Meta Tags and Configuration Variables* manual is a reference for those familiar with the basics of Tango.

CHAPTER TWO

# *Tango Editor Basics*

2

*An Explanation of the Tango Editor Interface and Some Common Operations*

This chapter helps you orient yourself to the Tango Editor interface and some of the common operations available to you.

The topics covered in this chapter include:

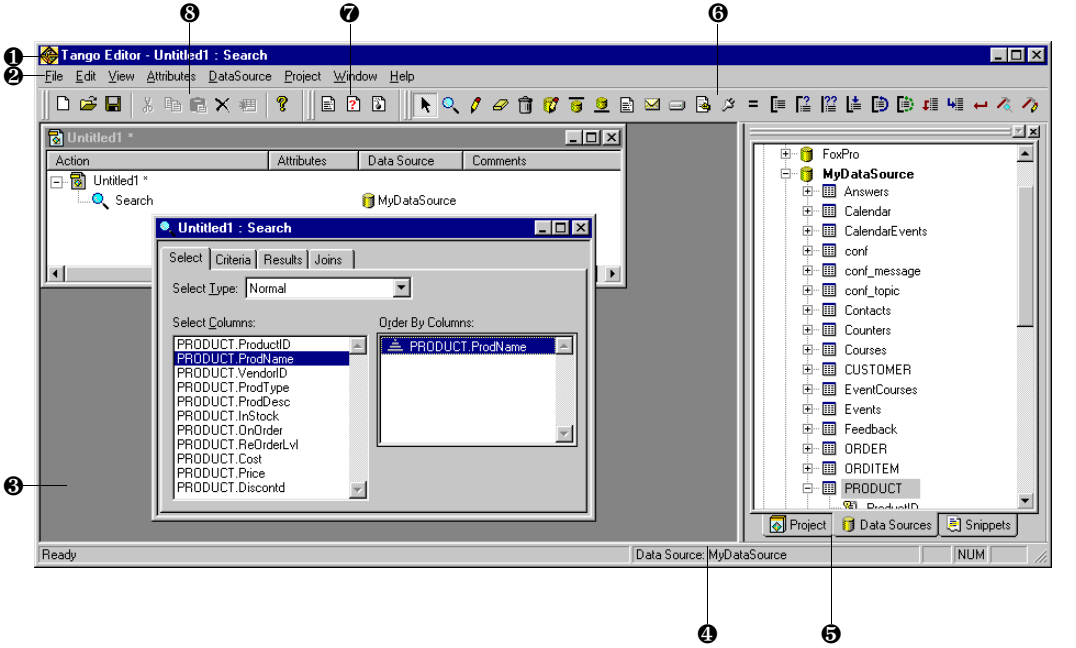
- Tango Editor window components
- using context-sensitive menus
- using HTML editing windows
- using Word Wrap
- using the SQL Query window
- finding and replacing text or regular expressions
- working with multi-column lists in action editing windows
- keyboard shortcuts.

# Tango Editor Window Components


To start Tango Editor, do either of the following:

- In the Tango folder, double click the Tango Editor icon.
- From the **Start** menu, select **Programs**, point to **Tango Enterprise 3.0,** then to **Tango Editor**.

The main Tango Editor window appears.




❶ The main title bar displays the Tango Editor name and the name of the current (front most) application file or the SQL Query window.

❷ The menu bar contains pull-down menus for Tango Editor commands. Click a menu title to open it, then click a command to select it. Commands appearing in gray do not apply to the operations you are trying to perform.

❸ The main window area displays one or more application file windows, action editing windows, attribute editing windows, or the SQL Query window.

❹ The Status Bar displays messages about the Tango Editor environment, such as when connecting to a data source, the currently connected data source, or when passing the cursor over a toolbar icon to display its name/function. It also shows if CAPS LOCK, NUM LOCK, and SCROLL LOCK on the keyboard are switched on.

❺ The Workspace includes tabs for Data Sources, Snippets, and Projects, if any exist. You switch among the three workspaces by clicking the corresponding tab.

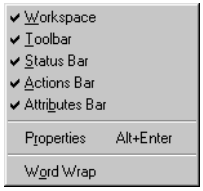
❻ Click icons on the Actions Bar and drag them into an open application file to add them to the file.

❼ Click icons on the Attributes Bar to assign attributes to selected actions.

❽ Click icons on the toolbar to select the main Tango Editor commands. For example, to save an application file, you can either choose **Save** from the **File** menu, or click the Save icon on the toolbar.

## Viewing Interface Objects

You can choose to show or hide the Workspace window or any of the toolbars by enabling the object's name from the **View** menu. A checkmark beside the name indicates the object is visible in the interface. Uncheck the name to hide the object.



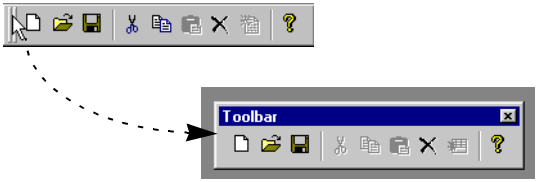

To hide the Workspace window, you can also right click it and click **Hide** from its context-sensitive menu.

## Floating and Docking Interface Objects

The Workspace window and toolbars are, by default, docked to the Tango Editor interface. You can drag them from the interface to undock, or float, them anywhere on your desktop. You can also dock them again.

To float an object on your desktop, simply click the undocking bars and drag the object to the desktop. If you want, you can then resize

the object. Position the cursor over the object's border, and when the cursor changes to the resize arrow, click and drag its border.




To dock the object to the interface again, drag it to anywhere in the toolbar area.

## Floating and Docking the Workspace Window

You can float the Workspace window in the Tango Editor main window or anywhere on your desktop, or dock it to the interface. To do this, you enable (checkmarked) or disable (unchecked) commands appearing in the Workspace window's context sensitive menu.

To float the Workspace window only in the Tango Editor main window, right click the Workspace window and click **Float in Main Window**. A checkmark appears beside the command indicating the option is enabled. This prevents you from dragging the Workspace window beyond the borders of Tango Editor.

If you want to drag the Workspace window to anywhere on your desktop, disable **Float in Main Window** and drag the Workspace window to another position.

To dock the Workspace window to the interface, drag the Workspace window to the toolbar area.

**Note** You cannot dock the Workspace window to the interface with the **Float in Main Window** option enabled.

To avoid inadvertently docking the Workspace window to the interface, right click the Workspace window and disable **Allow Docking**.

## Using Context-Sensitive Menus

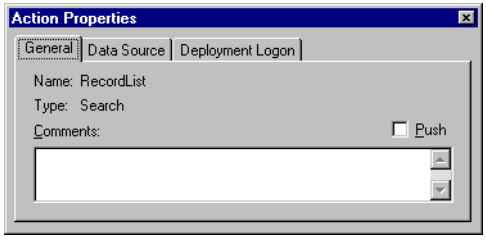
In most Tango Editor windows and dialog boxes, you can position the cursor on a particular area of the screen and click the right mouse button to display a context-sensitive menu of commands. The commands that appear relate to the object you clicked. Commands that appear grayed out are not applicable to the current object.

For example, another way to hide the Workspace window is to right-click it and select **Hide** from the context-sensitive menu.

## Properties Window

The Properties window allows you to view information about and add comments to a selected object. Selectable Tango objects include data sources (including tables and columns), application files, and actions. In general, the Properties window changes to show the properties of the currently selected object.

The following is an example of an Action Properties window for a Search action.



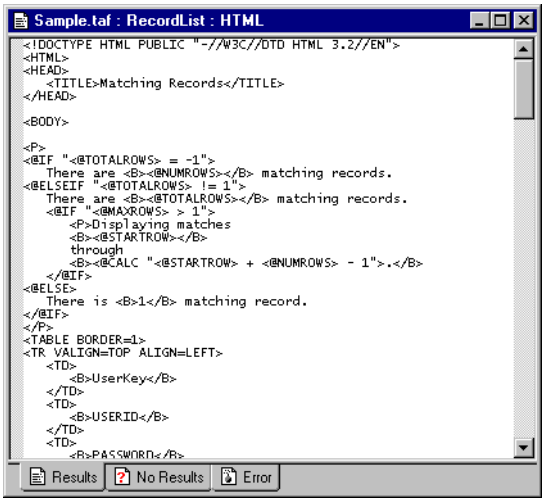

### *To open any Properties window*

1. Select the object you want to view information about.
2. Do either of the following:
   - From the **View** menu, choose **Properties**.
   - Right click the object, and choose **Properties** from the context-sensitive menu that appears.

The Properties window can be left open. Clicking an object with properties updates the window to show information about that object's properties.

# HTML Editing Window

Each action in an application file can have HTML associated with it. Whenever you open the assigned attribute of an action, the corresponding HTML editing window appears. You can create or edit any assigned HTML using this window. An example of an HTML editing window is as follows.




The title of the window follows the form:

```
<Document> : <Action> : <HTML>
```

This example shows the HTML editing window for the **Results HTML** of a Search action named “RecordList” within the `Sample.taf` application file.

Tango Editor supports the standard editing commands. The **Edit** menu displays the following editing commands when an HTML editing window is open.



You can also right click the HTML editing window to display a context sensitive menu at the cursor position in the window. The following table lists the commands available in the menu.

| Command | Function |
|---|---|
| Undo | Undoes the last change made to the text. |
| Cut | Removes the selected text from the window. |
| Copy | Copies the selected text to the clipboard. |
| Paste | Pastes text on the clipboard at the cursor position. |
| Delete | Deletes the selected text. |
| Select All | Selects all text within the HTML editing window. |
| Insert Meta Tag | Displays the Insert Meta Tag dialog box. For more information, see “Inserting Meta Tags” on page 113. |

Closing the editing window automatically saves any changes you make. To cancel any changes, you can choose **Undo**, or close the file without saving.

For more information, see "Setting Preferences" on page 303.

To make editing of your files easier and clearer, you can color code many of the HTML and text components that may appear—HTML, meta tags, attributes, default text, and comments.

You can also position text using tab characters. Tabs are stored as tab characters and are not converted to spaces. Tabs have no effect on the display of HTML in the Web browser; they are used to make the HTML that you enter more readable.

You specify the number of space characters that equal one tab character in the Preferences dialog box. You can also specify whether you want Tango to insert tab characters to start a new line at the same indent level as the previous line.

For more information on converting former query documents to Tango application files, see the *Getting Started Guide*.

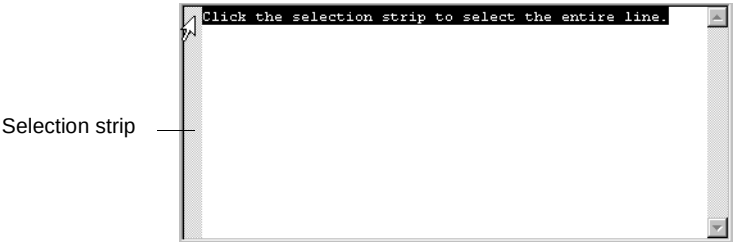
**Note** If you open an application file created in an earlier version of Tango, any <@COL> meta tags are automatically converted to <@COLUMN> meta tags.

You can enter any amount of text in an HTML editing window. You can also drag and drop text from elsewhere, for example, from other editing windows.

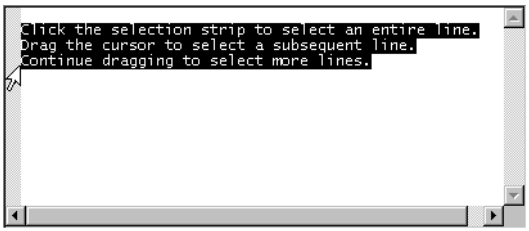
**Note** Word wrap is not available in the HTML editing window.

Selecting lines of text in an editing window is easy. You simply use the selection strip next to the line to select the entire line. When you select a line, it is highlighted. The following describes the operations you can perform using the selection strip.

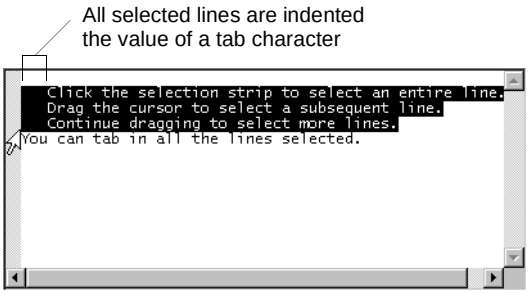
- To select a single line, click the selection strip beside the line you want to select. The entire line is highlighted.



- To select multiple lines, depress the mouse button in the selection strip next to the first line you want to select and drag the cursor up or down the selection strip to highlight subsequent lines.




To indent selected lines, press the TAB key. All selected lines are indented the number of characters equal to one tab character. Press the TAB key again to indent the selected lines farther; press the SHIFT+TAB keys to reduce the indention.

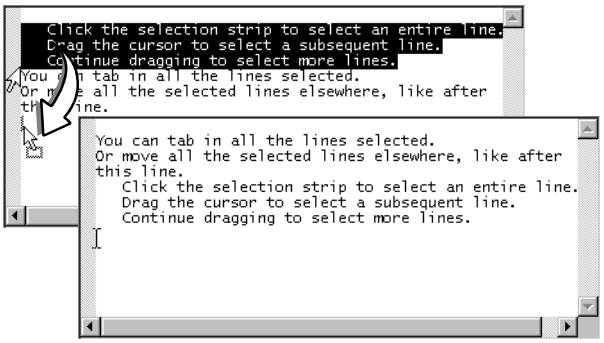



You set the number of characters equal to a tab character in the Preferences dialog box.

For more information, see “Using the Preferences Dialog Box” on page 304.

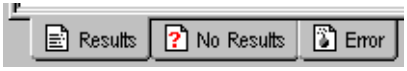
**Caution** You can only use the TAB key to indent the selected lines. Using the SPACE BAR instead replaces all the selected lines with a single space character.

- To move the selected lines elsewhere in the editing window, simply drag them to a new position. When you drag the selected lines, the cursor changes to . Release the mouse button where you want the selected lines to appear.




**Note** You can also use the standard editing commands (such as **Undo**, **Cut**, **Copy**, **Paste**, and **Delete**) on text selected using the selection strip.

The HTML editing window is common for any of the HTML attributes that may be assigned to an action—Results HTML, No Results HTML, and Error HTML. Only the applicable attribute tabs for the selected action appear at the bottom of the window.

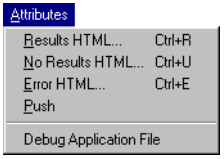



You switch among the HTML editing windows by clicking the appropriate tab.

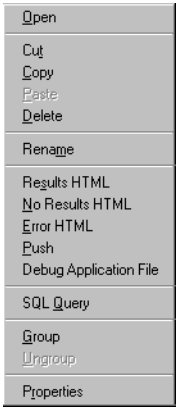
Only the applicable page or response HTML tabs appear at the bottom of the window.

You can also open the attribute HTML associated with an action by doing one of the following:

- Select an action icon / name, and choose the attribute type from the **Attributes** menu.



- Right click the action icon / name, and choose the attribute type from the context sensitive menu.

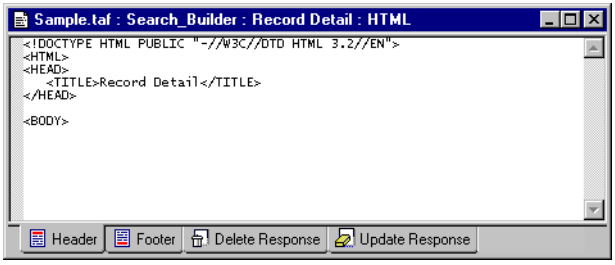



- Double click the attribute icon in the application file.

The corresponding HTML editing window opens.

For more information on customizing page and response HTML for the Search Builder, see “Configuring the Search Builder” on page 151.

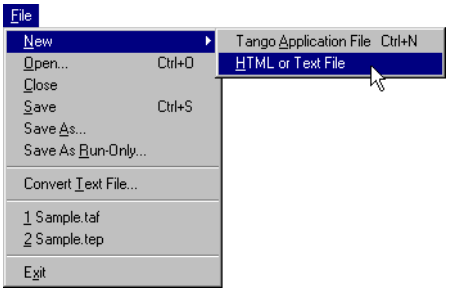
The Search Builder also uses HTML editing windows so you can customize page HTML (Header and Footer) and response HTML (Update Response, Delete Response, and No Results) for the Search, Record List, and Record Detail pages. For example, the following HTML editing window shows the Header HTML for the Search Builder’s Record Detail page in the `Sample.taf` application file.



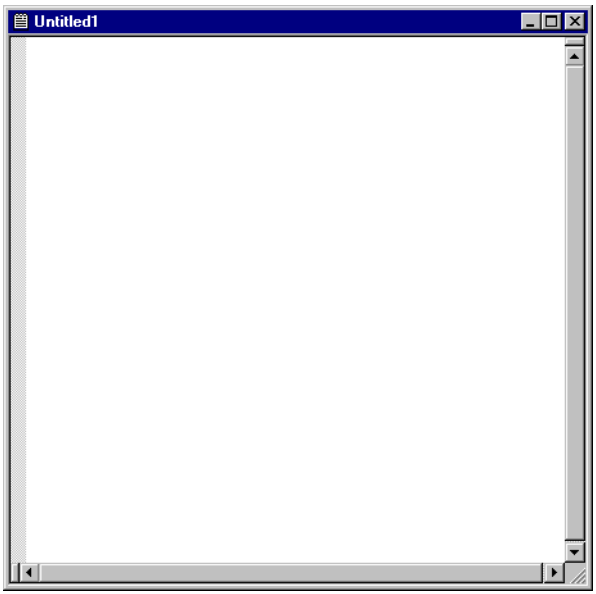

In addition to editing an action’s or Search Builder’s associated HTML, you can also use Tango’s editing capabilities to create and edit HTML and text files. The editing capabilities and window settings described for HTML action attributes apply equally to HTML and text files opened for editing with Tango Editor.

### *To create a new HTML or text file*

From the **File** menu, choose **New**, then **HTML or Text File** from the submenu.




A blank editing window opens.




The default window name is “Untitled1”, until you save it as another name. Subsequent new windows are named “Untitled*n*”, where *n* is the next number in the series, that is, the second window opened is “Untitled2”, and so on.

### *To save a new HTML or text file*

1. From the **File** menu, choose **Save** or **Save As**.

   The Save As dialog box appears.

2. In the **File name** field, type the name of your file and an appropriate extension. The default extension is `*.txt`.

   The **Save as type** drop down list includes file types: `*.html`, `*.htm`, `*.htx`, `*.txt`, `*.css`, `*.inc`, `*.log`, `*.xml`, `*.XSL`, `*.dtd`, `*.sql`.

When you save a text file and a project is open, Tango automatically asks if you want to add the saved file to the open project.

### *To open an HTML or text file*

1. From the **File** menu, choose **Open**.

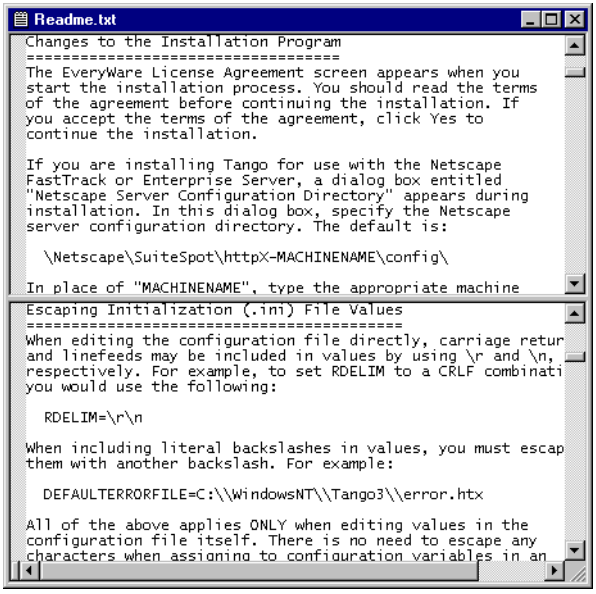
   The Open dialog box appears. The Open dialog box displays the same **Files of type** as listed in the table on page 64.

2. Select the file to open.

3. Click **Open**.

**Tip** You can also open a file of a supported type simply by dragging it from the Windows Explorer into Tango Editor or onto the Tango Editor icon, if Tango Editor is not already open.

You can also split the editing window into two or more panes. This allows you to see text from different locations in the same HTML or text file at the same time.



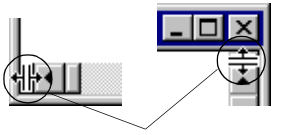

This example shows text in adjacent panes from two different places in the same `Readme.txt` file. This is useful if you want to see text from different places in the same file at the same time, or if you want to cut, copy, or paste text between the open panes.

**Note** Split information is not saved with a file.

### *To split the editing window into multiple panes*

1. Move the cursor over a split box.

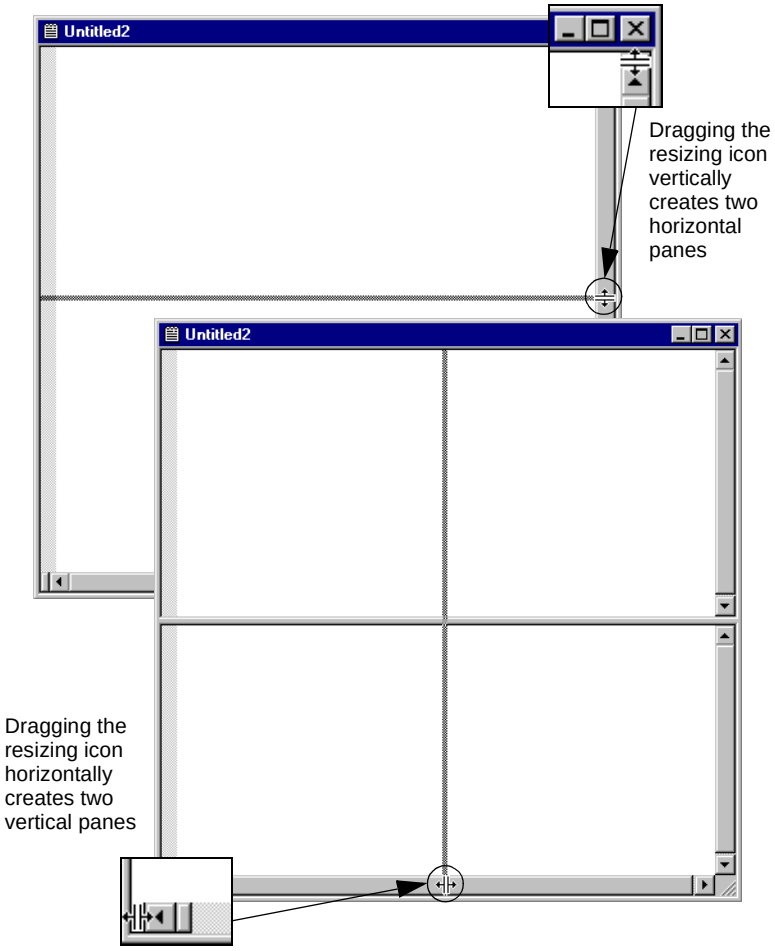
   A split box is located in the bottom left corner of the editing window and another in the upper right corner.



When you pass the cursor over the split box, the cursor changes to a resizing icon

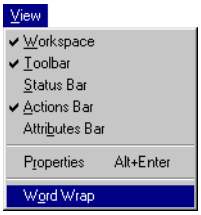
The cursor changes to a resizing icon.

**2** Drag the resizing icons to create separate window panes.



# Word Wrap

The **Word Wrap** command in the **View** menu is available for certain text windows.





When the command is available, selecting it enables or disables word wrap. A checkmark indicates word wrap is enabled.

If word wrap is disabled, a horizontal scroll bar is available to view text outside the boundaries of the text window.

Word wrap is available in the Direct DBMS action window, Script action window, and Mail action window, among others.

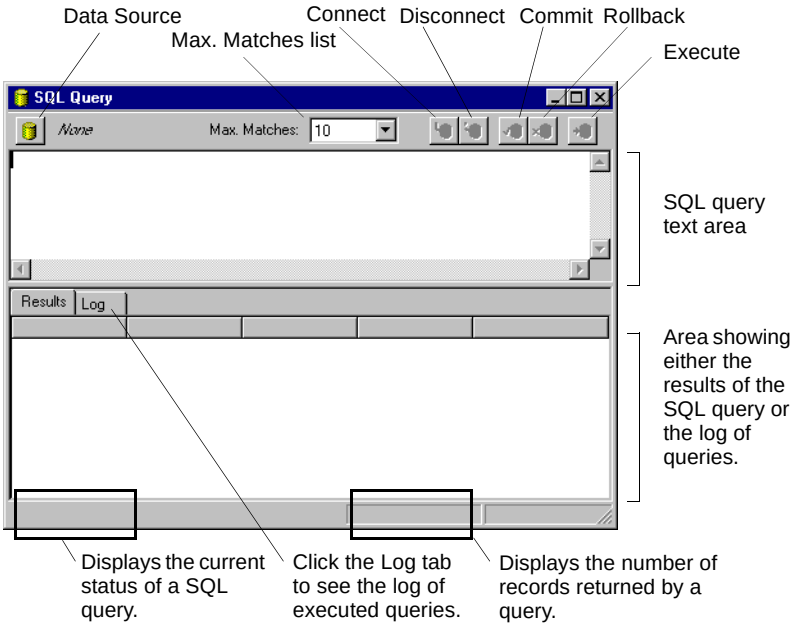
**Note** Word wrap is not available in HTML editing windows.

# The SQL Query Window

For more information, see “Performing a SQL Query” on page 38.

The SQL Query window gives you a convenient way of performing simple SQL queries within Tango Editor, for example, to test your Direct DBMS actions or to check database values.

The SQL Query window displays the following components.




If you want to resize the query and results areas, place the cursor over the areas separator to display the resizing icon.

Then click and drag it up or down to change the sizes of each area.

## Setting Up a SQL Query

The components and functions of the SQL Query window are as follows:

- **Data Source Button**. The data source you want to perform window operations against. When you first open the SQL Query window, the data source is set to **None**.

  If you change the data source assigned to the window, any existing connection closes.

- **Max. Matches**. The maximum number of records you want the SQL query to return. You can select from **1**, **10**, **25**, **50**, or **100**. The default is **10**.

For more information on SQL `COMMIT` and `ROLLBACK` operations, consult your SQL documentation.

- **Commit and Rollback Buttons**. Performs a SQL `COMMIT` or `ROLLBACK` operation on the assigned data source. `COMMIT` causes any changes made to the data source by the query to be saved. `ROLLBACK` causes any changes made by the query to be discarded.

  If you are not connected to a data source, these buttons are disabled.

- **Connect and Disconnect Buttons**. Connects to or disconnects from the current data source.

  When you try to connect without first assigning a data source, the Data Source Selection dialog box appears to select a data source from.

- **Execute Button**. Executes the SQL query in the query text area. If you are not connected to the data source when **Execute** is selected, the connection is made automatically.

  Any data returned by the SQL query appears in the **Results** area of the SQL Query window. If the **Results** area contains data and the current query returns no data, the **Results** area is cleared of any data.

  After execution, the connection to the data source remains open.

  To cancel an executed query, press the ESC key. If results are being returned when a cancel request is made, the **Results** area shows all the data returned to that point.

You can specify a different default font for text appearing in the SQL query text and results areas in the Preferences dialog box; see "Setting Preferences" on page 303.

For more information, see "Dragging Actions into SQL Query Text" on page 37.

- **Query Text Area**. Displays the SQL query text to be executed.

  The query text area supports standard cut, copy, and paste operations, including drag and drop. You can drag and drop tables and columns into the SQL Query text area from the Data Sources workspace.

  You can also drag any database action (except Transaction) from an application file to the SQL Query window to see the SQL Tango generates for it.

  If you select only part of the SQL when executing the query, only that part is sent to query the database.

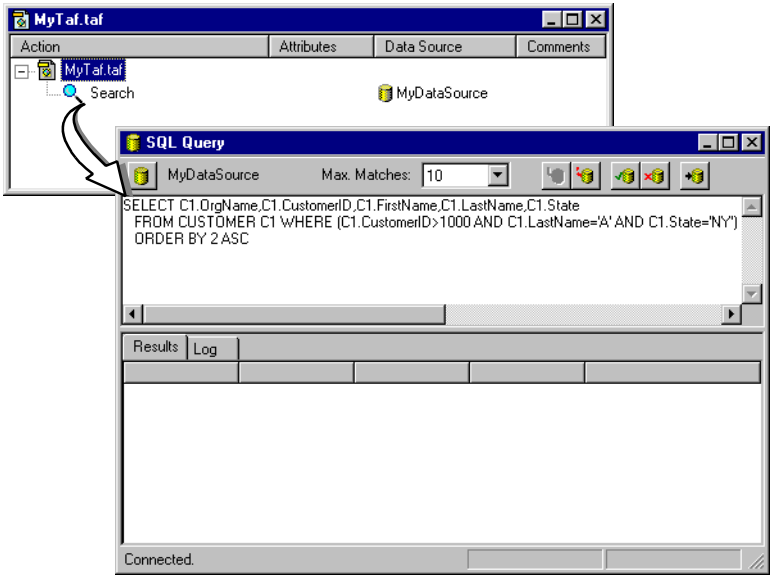
- **Results Tab**. Displays in columns and rows the results of the SQL query.
- **Logs Tab**. Displays the log of executed queries.

- **Status Area**. Shows the current status of the SQL query. The status messages appear as follows:

| Status | Description |
|---|---|
| Not connected | No connection is established. |
| Connecting... | Appears during connection to the data source. |
| Connected | Connection is established. |
| Executing... | Appears during execution of query. |
| Rolling back changes... | Appears during rollback operation. |
| Committing changes... | Appears during commit operation. |

## Dragging Actions into SQL Query Text

You can drag any database action, except a Transaction action, which does not generate SQL, from an application file into the SQL Query window.




When you do this, some SQL Query window attributes are set based on the contents of the action. The following attributes are automatically set:

- **Max. Matches** (for a search action) is set to the action's maximum matches value; otherwise, it is set to unlimited.

- The data source is set to the action's data source, and closes any existing database connection (if the data source is different from the current data source).
- The SQL text is the data source-specific SQL that Tango Server generates when the action is executed.

**Note** Any meta tags from the action are placed in the text as-is. It also does not include any text automatically added to the action's SQL by the server.

- The **Results** area is cleared of the currently displayed results.

## Performing a SQL Query

You can also choose the **SQL Query** command from a context-sensitive menu. Right click the application file window or an open action window.

### *To perform a SQL query*

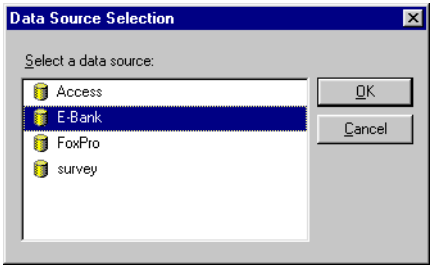
1. From the **Window** menu, choose **SQL Query**.

   An empty SQL Query window appears.

2. Click the **Data Source** button.

   When you first open the SQL Query window, the data source is **None**.

   The Data Source Selection dialog box appears.




3. Select the data source you want to perform SQL Query window operations against and click **OK** to load the tables and columns of that database.
4. From the **Max. Matches** drop-down list, select the maximum number of records to return from a SQL query: **1**, **10**, **25**, **50**, or **100**.

5. Click the **Connect** button to connect to the current data source.
6. In the SQL Query text area, enter the SQL query text to be executed.

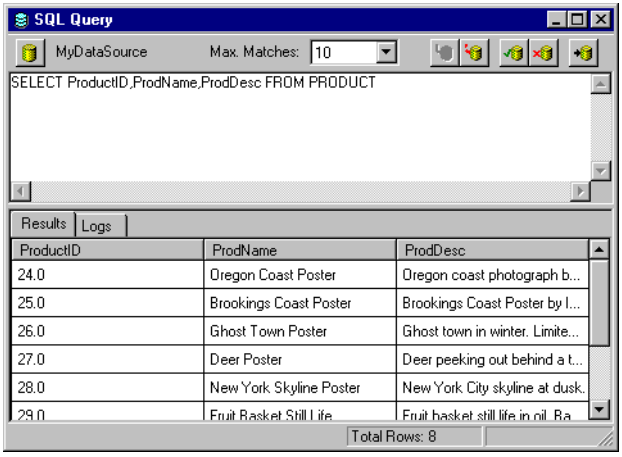
**7** Click the **Execute** button.

If you select part of the SQL in the SQL Query text area, only that part is executed when you click the button.

**8** If you want to perform a `COMMIT` or `ROLLBACK` operation on the assigned data source, click the corresponding **Commit** or **Rollback** button.

The results of the SQL query, if any, appear in the **Results** area.

The following is an example of SQL query text and the returned results.



# Using Tango Application Files

A Tango application file provides a powerful and flexible means for you to construct dynamic applications that run on your Web server and that interact with databases, other applications, and users running Web browsers. They are like programs or scripts in that they determine what operations Tango Server performs. Tango Server provides the brains, but it does nothing without the specific instructions you provide in the form of application files.

You add actions to an application file. When Tango Server runs the application file, it generates the HTML that is used by the browser to display the forms required to allow interaction with databases and other applications.

You can use the Search Builder and New Record Builder to have Tango Editor build search and insert record applications for you.

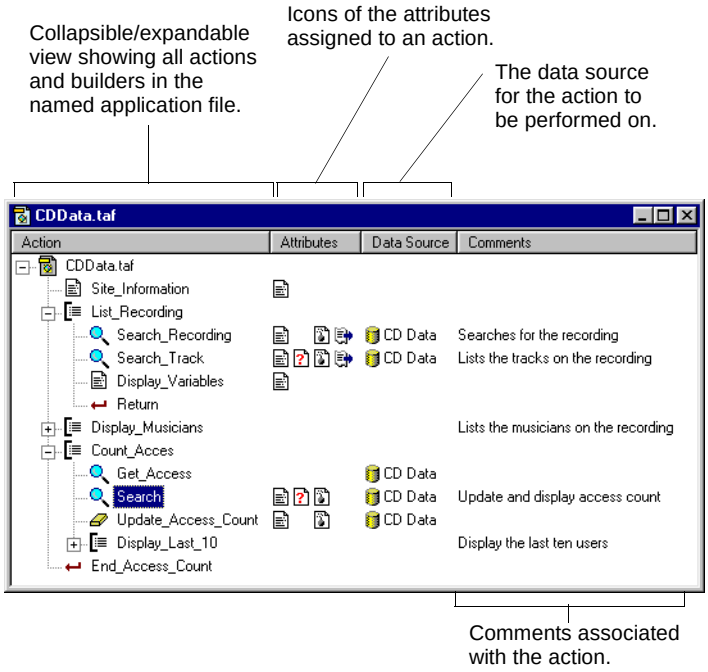
An *application file* is a file containing a series of Tango actions that when executed by Tango Server generates HTML and controls interaction with databases and other applications.

## Application File Window

In Tango Editor, whenever you open an application file, the application file window shows you the following information:

- action icons and names, including those for builders, in the order Tango Server executes them (unless a control action redirects the flow of the execution)
- attributes assigned to an action, if any
- data sources for all database actions
- any associated comments.

The application file window also includes icons for attributes and data sources. The following diagram shows a typical application file window and its components.



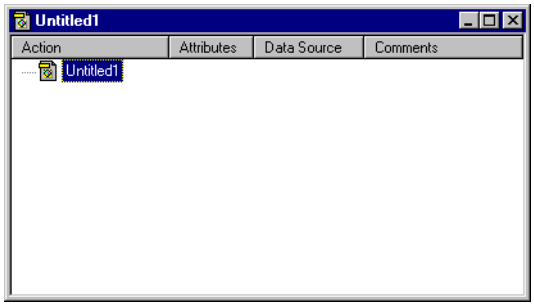

## Creating an Application File

New

***To create a new application file***

1 From the **File** menu, choose **New**, or in the toolbar click the **New** icon.

An untitled application file opens.



## Dragging Columns

When creating or modifying a Tango application file and actions, you must specify which database columns to use in various places. To do this, you drag the columns from the Data Sources workspace to the appropriate place in the file.

| To ... | Do This ... |
|---|---|
| Select contiguous columns | Click the first column you want to select and SHIFT click the last one. |
| Select discontiguous columns | CONTROL click each of the desired columns. |
| Select all columns in a table | Drag the table name to the file. |

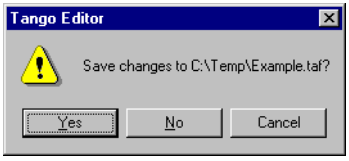
To see the Data Sources workspace, click the Data Sources tab. A workspace appears containing information about data sources, such as the currently defined data sources and all tables and columns. If no data sources are set up yet, only the data source types appear.

## Saving an Application File

### *To save an application file*

1 From the **File** menu, choose **Save**.

If the application file has never been saved, the Save As dialog box appears.




If it has been saved previously, Tango Editor saves it using the existing name and location.

2 Navigate to the desired location for the application file.

See “Executing Application Files” on page 45

For Tango Server to execute the application file it must be located in or below the Web server’s document root directory.

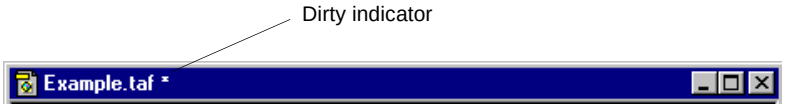
3 Name the application file.

Tango application file names generally end in `.taf`. This is the standard suffix used to identify files that Tango Server should execute. With the plug-in, this extension is required, unless you change the mapping of extensions in your Web server. Consult your Web server documentation for details. With the CGI

version of Tango Server and a suffix-mapping you set up yourself, you may use any extension you like. The `.taf` extension is added if no extension is specified.

4 Choose **Save**.

Whenever you change an application file, and the file has not been saved, an asterisk appears beside the file name. This asterisk is called a *dirty* (unsaved changes) indicator.




Once you save the application file, this indicator disappears.

## Saving an Application File as Run-Only

Run-only application files can be executed by Tango Server, but they cannot be opened by Tango Editor.

Saving an application file as run-only allows you to create and distribute packaged Tango solutions while preventing users from editing the actual application file. Run-only application files are executed and referenced by Tango Server in the same way as editable application files. Saving an application file as run-only does not make its execution any faster.

**Caution** You cannot edit a run-only copy of an application file, and there is no way to make a run-only file editable. Make sure you keep an editable copy of any run-only application file.

### *To make an application file run-only*

1 With an application file open in Tango Editor, choose **Save As Run-Only** from the **File** menu.

The Save As dialog box appears.

You are saving a copy of your application file as run-only. Your original application file is not changed.

2 Name the run-only application file.

**Tip** You may want to give the run-only versions of your application files a special name to identify their type, such as `CustomersRO.taf`, where "`RO`" represents run-only.

3 Click **Save**.

A run-only version of the application file is saved in the location you specified.

**Note** If you are distributing your Tango solution, your customers need to purchase Tango Server. Alternatively, you can license Tango Server for distribution with your solution. Contact Sales at Pervasive for more information.

## Debugging Application Files

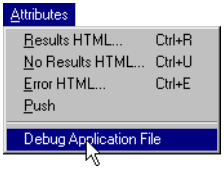
Setting the debug mode in Tango Editor lets you see useful information about your application file execution in your Web browser application.

### *To set the debug mode*

1 Open the application file you want debug information on.

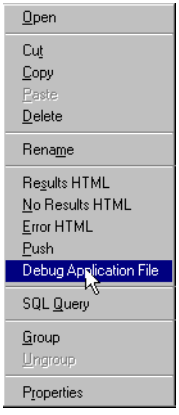
2 Do one of the following:

- From the **Attributes** menu, select **Debug Application File**.

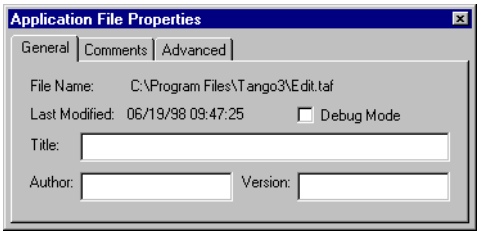



  A checkmark beside the command indicates the debug mode is enabled.

- Right click the application file window, and select **Debug Application File** from the context sensitive menu.



- From the **View** menu, choose **Properties**. Then select **Debug Mode** in the Application File Properties dialog box that appears.


When you execute the application file, debugging information appears at the bottom of the results returned. The debugging information shows information such as:

- arguments passed in (search and post arguments)
- actions executed
- values of variables
- SQL generated by database actions
- warnings (such as references to missing arguments).

For more information, see “Configuring Tango Server” on page 317.

You can force debugging for all application files by changing the `debugMode` system configuration variable using the application file that allows you to set system configuration (`config.taf`).

## Executing Application Files

Application files are executed in the same way HTML files are viewed—simply by specifying the name of the file in a URL. For example:

```
http://www.yourserver.com/shop/additem.taf
```

This example executes an application file called `additem.taf` located in the `shop` folder on the server `www.yourserver.com`.

**Note** This example assumes you are running one of the Tango Web server plug-ins or extensions (available for Microsoft’s Internet Information Server and Netscape’s Web servers). If you are using the Tango CGI instead, the syntax required may be different. See the *Getting Started Guide* for details.

You can pass parameters to the application file by using *search* arguments. These are name-value pairs appearing after a question mark in the URL. For example:

```
http://www.yourserver.com/shop/
additem.taf?item_num=8580
```

In this example, the `item_num` search argument has a value of "8580".

See "Assigning Variables With the Assign Action" on page 130

There are other ways of passing values to Tango application files. *Form* fields (post arguments) and *cookies* are two examples.

# Finding and Replacing Text

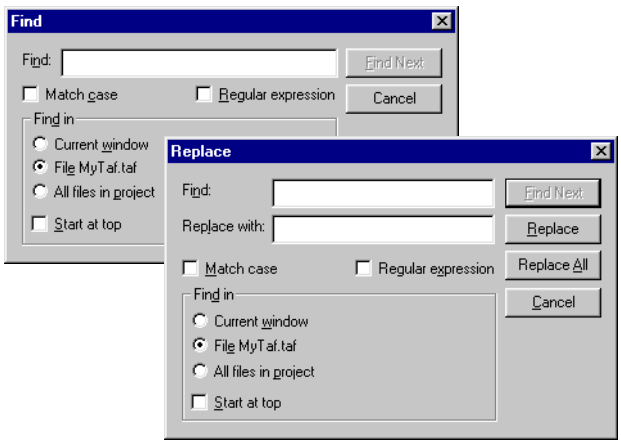
For more information, see “Using Regular Expressions” on page 50.

In Tango Editor, you can perform operations to find, or to find and replace text in application files. Tango Editor can perform both normal searches and searches using regular expressions.

## Performing Find Operations

For the purpose of this discussion, the term *string* refers to both character strings (that is, text) and regular expressions. You specify that the search is to treat the string in the **Find** field as a regular expression by selecting the **Regular expression** option in the Find or Replace dialog box.

If you want to find a certain string, you specify that string in the Find dialog box. If you want to find a certain string and replace it with another string, you do that in the Replace dialog box.




You can find any string that can be entered in any non-modal Tango Editor window. This includes values in criteria lists, action parameters you have entered—such as for the **Limit to** field in a Search action’s Results window, custom SQL, If action conditions, External action parameters, custom column definitions, and HTML. Tango Editor cannot find a string you did not explicitly enter, for example, data source names, user names or passwords entered by users, column names in Select lists, and join information.

You can perform find, and find and replace operations in open application files, action editing windows, HTML editing windows and projects. Unless specified otherwise, Tango Editor begins

searching at the insertion point indicated by the cursor and continues to the end of the search range specified in the **Find in** section of the dialog box.

### *To find, or find and replace a string*

1. Depending on the operation you want to perform, choose either **Find** or **Replace** from the **Edit** menu.

   The corresponding Find or Replace dialog box appears.

2. Specify your find or find and replace options as follows:

   - **Find**. Enter the string you want to find.
   - **Replace with**. Enter the string you want to replace the string in the **Find** field with.
   - **Match case**. If you want to perform a case-sensitive search, select the **Match case** option; otherwise, Tango Editor searches for a match irrespective of letter case. For example, a search for "customer" would find all instances of "customer", "Customer", and "CUSTOMER".
   - **Regular expression**. If you want to search for the string as a regular expression, you must select the **Regular expression** option. Otherwise, a normal search is performed.
   - **Find in**. You specify the search range in this area of the dialog box.

     **Current window**. Select this option to perform the find or replace operation in the window active at the time you chose the **Find** or **Replace** command. If you have a string selected in the active window, it automatically appears in the **Find** field.

     **File *filename***. Select this option to perform the find or replace operation in the file specified by *filename*. The name of the currently active file automatically appears as *filename*.

     **All files in project**. If you have a project open, this option is enabled. Select this option to perform the find or replace operation in all the files of the active project. If you have another application file open at the same time, which is not part of the project, Tango Editor excludes it from the find or replace operation.

- **Start at top**. Select this option to start the find or find / replace operation at the top of the search range specified in the **Find in** section. If this option is not selected, Tango Editor performs the search starting from the current cursor position.

**Note** If a search range is specified and the current cursor position is not within that range, the current cursor position is ignored and the search starts at the top of the specified range.

- **Find Next**. Select this button to start the search for the string specified in the **Find** field from the specified starting position.
- **Replace**. Select this button to replace the string specified in the **Find** field with the string specified in the **Replace with** field. Following the replace operation, Tango Editor automatically searches for the next instance of the find string.

  You can undo the last replace performed by selecting **Undo** from the **Edit** menu.
- **Replace All**. Select this button to replace automatically *all* instances of the string specified in the **Find** field with the string specified in the **Replace with** field. There is no confirmation of any replacement; however, at the end of the operation Tango Editor displays a message telling you the number of replacements made.

**Note** You cannot undo the Replace All operation. You can, however, choose to close a file without saving the changes to return it to its former state.

  If the search range involves several objects, those objects in which replacements are made are opened so you can save or discard the changes.
- **Cancel**. Select this button to stop the find or find / replace operation and to close the dialog box.

## Using Regular Expressions

A *regular expression* is formed by one or more special characters that represent a string of text.

**Note** To find a special character, precede it with a backslash, for example, * finds the asterisk (*) character.

### *To find any single character*

A period (.) finds any character except a newline character.

| Expression ... | Finds ... |
|---|---|
| **.use** | **fuse** but not **house** |

### *To repeat expressions*

Repeat expressions with an asterisk (*) or a plus sign (+).

A regular expression followed by an asterisk finds zero or more occurrences of the regular expression. If there is any choice, Tango Editor chooses the longest, left-most matching string in a line.

A regular expression followed by a plus sign finds one or more occurrences of the one-character regular expression. If there is any choice, Tango Editor chooses the longest left-most matching string in a line.

| Expression ... | Finds ... |
|---|---|
| **a+b** | **ab** and **aab** but not **a** or **b** |
| **a*b** | **b**, **ab**, and **aab** but not **baa** |
| **.*use** | **use**, **mouse**, and **paint the house**, but not **chair** |

### *To group expressions*

If an expression is enclosed in parentheses, ( ), Tango Editor treats it as one expression and applies an asterisk or plus sign to the whole expression.

| Expression ... | Finds ... |
|---|---|
| **(ab)*c** | **abc** but not **aabbcc** |
| **(.a)+b** | **xab** but not **b** |

### *To choose any character from many*

A string of characters enclosed in square brackets, [ ], finds any one character in that string. If the first character in the brackets is a caret (^), it finds any character except those in the string.

| Expression ... | Finds ... |
|---|---|
| **[abc]** | **a**, **b**, or **c**, but not **x**, **y**, or **z** |
| **[^abc]** | **x**, **y**, or **z**, but not **a**, **b**, or **c** |

A minus sign (-) within square brackets indicates a range of consecutive ASCII characters. For example, `[0-9]` is the same as `[0123456789]`. The minus sign loses its special meaning if it is the first character (after an initial caret, if any) or last character in the string.

If a right square bracket is immediately after a left square bracket, it does not terminate the string; however, it is considered to be one of the characters to match. If any special character—such as the backslash (\), asterisk (*), or plus sign (+)—is immediately after the left square bracket, it does not have its special meaning and is considered to be one of the characters to match.

| Expression ... | Finds ... |
|---|---|
| **[aeiou][0-9]** | **a9** but not **ae** |
| **[^bm]ate** | **date** but not **bate** or **mate** |
| **END[.]** | **END.** but not **END;** |

### *To find the beginning or end of a Line*

You can specify that a regular expression find only the beginning or end of the line.

If a caret (^) is at the beginning of the entire regular expression, it finds the beginning of the line.

If a dollar sign ($) is at the end of the entire expression, it finds the end of the line.

If an entire expression is enclosed by a caret and dollar sign (for example, `^the end$`), it finds an entire line.

| Expression ... | Finds ... |
|---|---|
| **^(the house).+** | **the house guest** but not **paint the house** |
| **.+(the house)$** | **paint the house** but not **the house guest** |

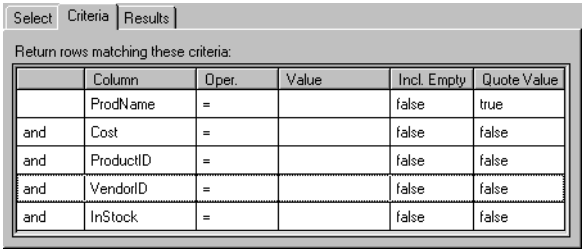
### *To re-use a regular expression in the Replace field*

Tango extends the regular expression functionality and allows you to remember and recall a part of a regular expression. Enclose the part to remember with parentheses. To recall it, use *n*, where *n* is a digit that specifies which expression in parentheses to recall. Determine *n* by counting occurrences of ( from the left. You can only use this feature in the **Replace** field of the dialog box.

**Tip** For more information on constructing POSIX regular expressions, ask your local UNIX guru, consult the FreeBSD regex man page, or try doing an Internet search for the term “POSIX 1003.2“.

# Working With Multi-column Column Lists

Many Tango actions include multi-column lists for entering parameters—the criteria list in the Search action, for example. This section describes basic techniques for working with these lists.




| | Column | Oper. | Value | Incl. Empty | Quote Value |
|---|---|---|---|---|---|
| | ProdName | = | | false | true |
| and | Cost | = | | false | false |
| and | ProductID | = | | false | false |
| and | VendorID | = | | false | false |
| and | InStock | = | | false | false |

### *To select an entire row*

Click the row's Column cell.

| | Column | Oper. | Value | Incl. Empty | Quote Value |
|---|---|---|---|---|---|
| | ProdName | = | | false | true |
| and | Cost | = | | false | false |
| and | ProductID | = | | false | false |
| and | VendorID | = | | false | false |
| and | InStock | = | | false | false |

### *To move a row*

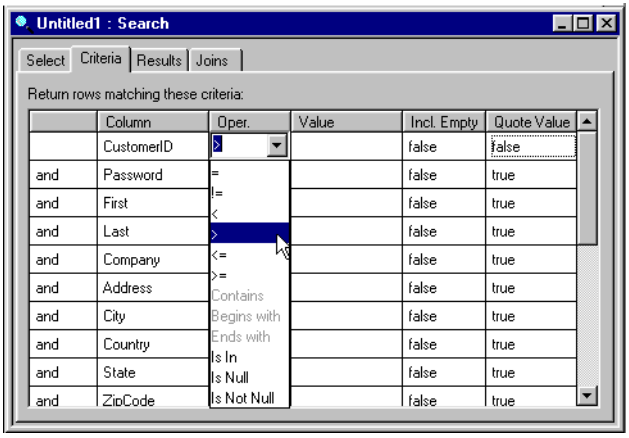
Select the row and drag it to the desired location.

| | Column | Oper. | Value | Incl. Empty | Quote Value |
|---|---|---|---|---|---|
| | ProdName | = | | false | true |
| and | Cost | = | | false | false |
| and | ProductID | = | | false | false |
| and | VendorID | = | | false | false |
| and | InStock | = | | false | false |

A flashing grey line indicates where the row is inserted when the mouse button is released.

### Drop-down Menus

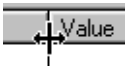
Various columns have drop-down menus in each cell. Place the cursor in the cell and click the mouse. A downward direction arrow appears. Click the arrow and the drop-down menu appears.




From a cell's drop-down menu, you can select from preset values (like "and / or" and "true / false").

### To resize a column

Click at the edge of the column in the list's header, and drag.




A double-headed arrow appears when you move your cursor between columns. Drag to resize the column.

To resize a column to fit the data in it, double click its right edge in the header.

### To delete a row

1. Select the row to delete.
2. From the **Edit** menu, choose **Delete**, or press the DELETE key on the keyboard.

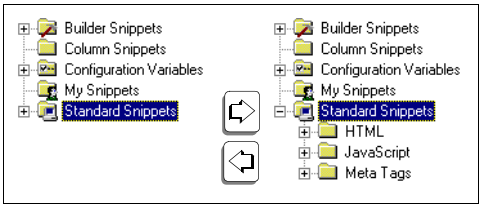
# Keyboard Shortcuts

The keyboard shortcuts, as they appear in Tango Editor menus, are as follows:

| Menu | Command | Shortcut |
|---|---|---|
| File | New | Ctrl+N |
| | Open | Ctrl+O |
| | Save | Ctrl+S |
| Edit | Undo | Ctrl+Z |
| | Cut | Ctrl+X |
| | Copy | Ctrl+C |
| | Paste | Ctrl+V |
| | Insert | Ins |
| | Delete | Del |
| | Select All | Ctrl+A |
| | Find | Ctrl+F |
| | Replace | Ctrl+H |
| | Rename | Ctrl+Enter |
| | Group | Ctrl+G |
| | Ungroup | Shift+Ctrl+G |
| | Insert Meta Tag | Ctrl+M |
| View | Properties | Alt+Enter |
| Attributes | Results HTML | Ctrl+R |
| | No Results HTML | Ctrl+U |
| | Error HTML | Ctrl+E |
| DataSource | Reload | F5 |
| Window | SQL Query | Ctrl+Q |
| Help | Help Home Page | F1 |

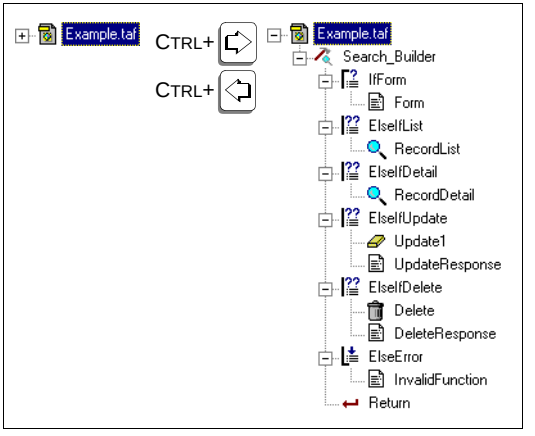
When working in the Project, Data Sources, and Snippets Workspaces, or in the application file window, you can expand and collapse any parent object by one level using the left and right keyboard cursor keys. A *parent* object is any object denoted in the view by the plus sign (⊞, expandable) and negative sign (⊟, collapsible).

- To expand the selected parent one level, press (right cursor key).
- To collapse the selected parent one level, press (left cursor key).




You can also use keyboard shortcut keys in an open application file window to expand and collapse the parent object through all levels at one time.

- To expand the selected parent, press CTRL+ .
- To collapse the selected parent, press CTRL+ .



CHAPTER THREE

# 3 Using Projects and Source Control

*The Basics of Tango Projects and Managing Files Using Source Control*

A *project* is a logical grouping of folders and files. It gives you the convenience of being able to organize your work for like sets of files, including application, HTML, and text files—in fact, for any type of file. Projects exist in Tango Editor only and do not interact with Tango Server.

Tango Editor can conveniently access popular source control systems, such as Microsoft® Visual SourceSafe™, INTERSOLV® PVCS®, and StarBase® Versions®. You can manage all your files from the Project Workspace, without having to launch your source control system separately.

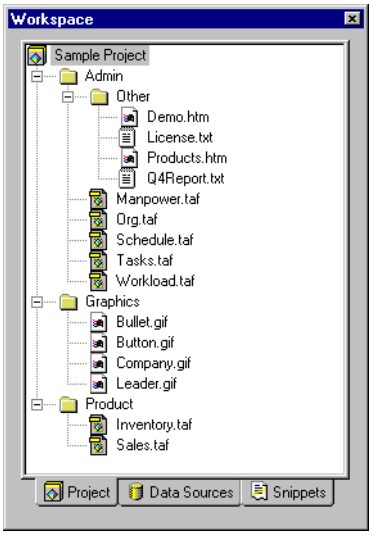
This chapter covers the following topics:

- working with Tango projects
- using source control in Tango.

# Working With Projects

When you create a new project, a Workspace opens displaying the project, including the project icon with the name of the project. The actual project name is the file name you assigned to the project prefixed to the word “Project”.

When you add a folder or file to a project, it is added alphabetically to the project list.




The project file contains information on the project, including a listing of the folders and files in the project.

**Note** The project file does not contain the actual files. It is just a listing to help you manage your projects.

## Performing Project Operations

You perform operations on the project file separately from the folders and files it contains. Path names of files stored in the project file are stored relative to the project file's location.

For more information on setting Tango Editor preferences, see "Setting Preferences" on page 303.

You can open any file appearing in the Project Workspace simply by double clicking the file name. The file automatically opens and displays its contents in the application defined by its Windows suffix mapping. Application files automatically open in Tango Editor; if you set Tango Editor as the default editor in the Preferences dialog box, HTML and text files also open in Tango Editor.

For more information on files under source control, see "Using Source Control in Tango" on page 67 and "Opening a File Under Source Control" on page 79.

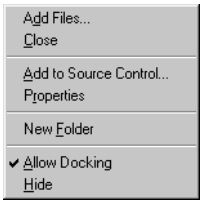
If you try to open an application file that is currently under source control and not checked out, Tango Editor prompts you to check it out first.

You can also conveniently execute certain project commands directly in the Workspace. Right clicking the icon or name for a project, folder, or file displays a menu of applicable commands and Workspace window commands.

**Note** Source control commands only appear in Tango Editor if you have a source control system installed on your machine.

**Project**
Right clicking the project name or icon displays a menu of commands

When project is not source controlled




When project is checked in


When project is checked out


**Folder**
Right clicking the folder name or icon displays a menu of commands

When folder is checked in


When folder is checked out


**File**
Right clicking the file name or icon displays a menu of commands

When file is checked in


When file is checked out

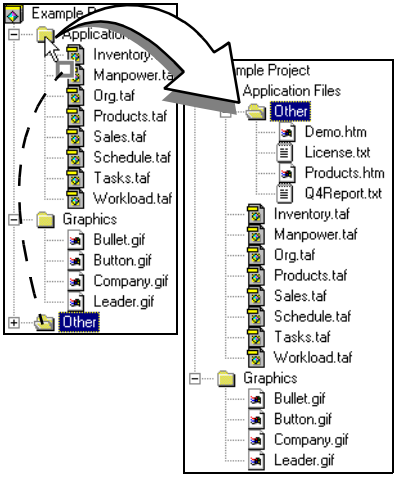

For more information, see “Finding and Replacing Text” on page 47.

One of the powerful editing features of Tango is its ability to find and replace text in all the files—application, HTML, and text—of a project. The project must be open for the find and replace operation to take place in the applicable files of the project; all non-text files are ignored. If Tango finds the specified text string, it automatically opens an editing window showing the corresponding file or HTML attribute for an application file.

You can move folders and files within the Project Workspace simply by dragging them to a new location.

- Dragging a file to a folder adds that file to the target folder.
- Dragging a folder to another folder makes it—and any files in it—a subfolder of the target folder. The following diagram

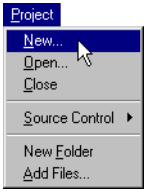
shows the `Other` folder being dragged to the `Application Files` folder. The `Other` folder and the files in it then appear under the `Application Files` folder.



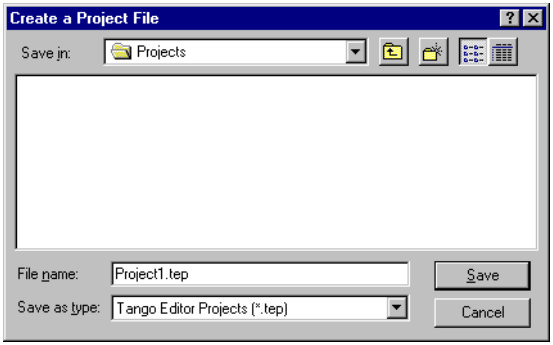

## Creating a New Project

***To create a new project***

**1** From the **Project** menu, choose **New**.



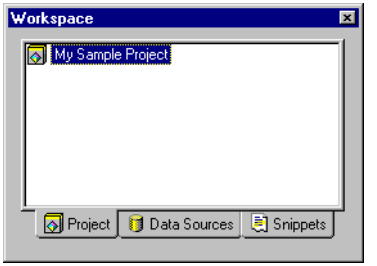

The Create a Project File dialog box appears.



2 Specify a project file name and location.

Project file names end in ".tep". This is the standard suffix used to identify the file that lists the folders and files forming a project.

3 Choose **Save**.




The project name appears in the Project Workspace.

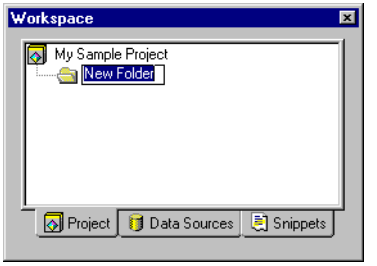
## Adding a Folder to a Project

### *To add a folder to an open project*

Do one of the following:

- From the **Project** menu, choose **New Folder**.
- Right click the project icon or name to display a context sensitive menu, and choose **New Folder**.
- Drag an existing folder from the Windows Explorer.

If you are adding a new folder, a folder icon named "New Folder" appears below the project icon. The folder name is automatically selected for easy renaming. The folder name must be unique at the level you are adding the folder.




You can drag a folder from the Windows Explorer into the project at the root level or into any existing project folder. The folder and any files it contains are added at that location.

**Note** If the folder being dragged contains subfolders, Tango ignores them. You must add the subfolders to the project one folder at a time.

You cannot add a folder to a project using the same name at the same level as an existing folder. In other words, if you have a folder named `Graphics` at the first level below the project root, you cannot add another `Graphics` folder at that first level. You can, however, add a `Graphics` folder below another `Graphics` folder.


To rename a project folder, right click the folder icon or name, and choose **Rename** from the context sensitive menu.

Folder names appear alphabetically in the Project Workspace.

## Adding Files to a Project

You cannot add a file with the same name as an existing file in the same location in the project list. You can, however, add a file with the same name as an existing file in a different location.

File names appear alphabetically in the Project Workspace.

**Note** The order of application files in the project list has no bearing on the order that Tango Server executes them.

For more information on setting source control preferences, see “Source Control” on page 309.

If you enable the **Prompt to add files when inserted into a project** option in Tango Editor’s source control preferences, you are prompted to add the files to source control. Click **Yes** to add the files or **No** to cancel.

### *To add files to the project root*

You can also drag a file or multiple files from the Windows Explorer into the Project Workspace.

1 Open the project, and do one of the following:

- From the **Project** menu, choose **Add Files**.
- Right click the project name, and choose **Add Files** from the context sensitive menu.

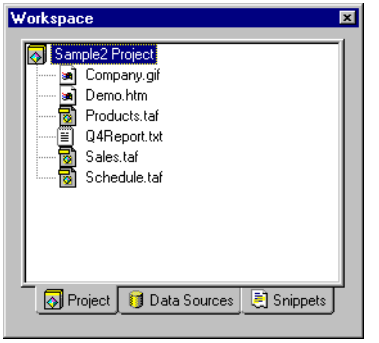
A file selection dialog box appears. The types of file selection supported include the following:

| Type | File Extension |
|---|---|
| Tango Application File | *.taf, *.qry |
| HTML and Text Files | *.html, *.htm, *.htx, *.txt, *.css, *.inc, *.log, *.xml, *.XSL, *.dtd, *.sql |
| Graphics Files | *.gif, *.jpg |
| All Files | *.* |

2 Select the files you want to add to the project.

3 Choose **Open**.

The added files appear in the Project Workspace.




### *To add files to a folder*

You can also drag a file or multiple files from the Windows Explorer into the Project Workspace.

1 Open the folder.

2 Do either of the following:

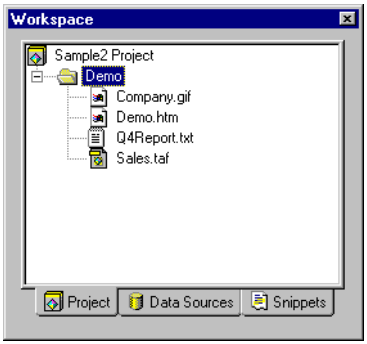
- From the **Project** menu, choose **Add Files**.
- Right click the folder name and choose **Add Files to Folder** from the context sensitive menu.

A file selection dialog box appears. See the table of supported file types on page 64.

3 Select the files you want to add to the folder.

4 Choose **Open**.

The added files appear in the open folder.




## Removing Files and Folders From a Project

***To remove files and folders from an open project***

1 Select the files or folder you want to remove from the project.

**Note** If you select a folder, you remove it and any of the folders or files it contains.

2 Do either of the following:

- Right click the files or folder, and choose **Remove** from the context sensitive menu.
- Press the DELETE key.

A message appears asking you to confirm that you want to remove the selected item(s).

3 Choose **Yes**.

Removing a file from a project does not delete the file. The file remains intact so you can use it again or add it to another project.

## Closing and Opening a Project

To close an open project, choose **Close** from the **Project** menu.

To open an existing project, choose **Open** from the **Project** menu. If another project is already open, Tango first closes it and then opens the selected project. Any changes you made to the project being closed are automatically saved.

When you open a project, the last view state is restored, that is, folders appear expanded or collapsed as they did previously.

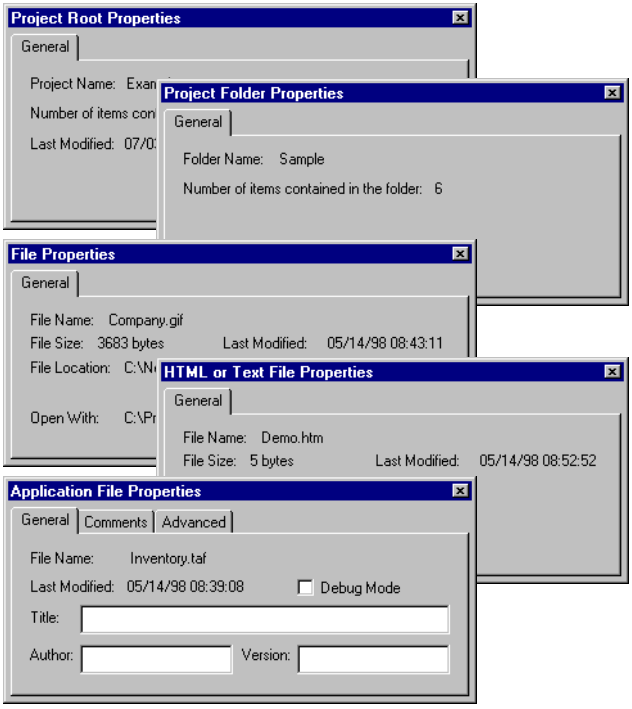
Any changes you make to an open project are automatically saved as you make them.

## Properties of Project Files

You can see the properties for any object in the Project Workspace: project, folder, and file.

Right click a Workspace object, and choose **Properties** from the context sensitive menu.

The properties information associated with each project object is as shown in the following examples.




## Editing HTML and Text Files

In addition to application files, a project file can include any other type of file. For HTML and text files, Tango has built-in editing capabilities. (See "HTML Editing Window" on page 24.)

For more information on setting Tango Editor preferences, see "Setting Preferences" on page 303.

When you open any file included in a project that has an extension listed in the table on page 64, Tango's HTML editing window opens (if you enable the **Open text files in projects using Tango Editor** option in Tango Editor's source control preferences). Otherwise, Tango launches the **Opens with** application you have specified in the Windows Explorer for that file type.

If a project is open when you save an HTML or text file in Tango, you are automatically asked if you want to add the file to the current project. Click **Yes** to add the file to the project root or **No** to cancel.

# Using Source Control in Tango

For more information on the file types that Tango projects support, see the table on page 64.

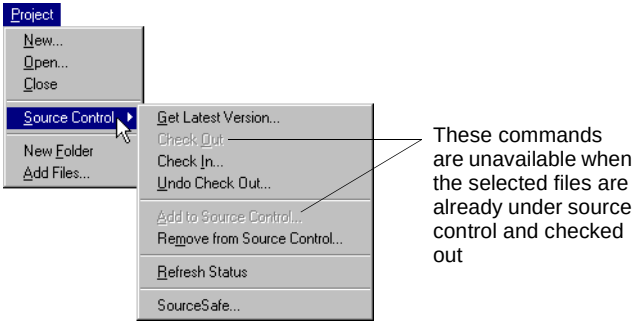
Tango Editor supports the common source code management features for all files incorporated into a Tango Editor project.

**Note**

- Tango Editor supports any source control system that conforms to Microsoft's Source Code Control API, also used by Microsoft's Visual Studio development environment. In other words, if a particular source control system works with Visual Studio, it works with Tango Editor.
- If your source control system installation has an option to integrate with Visual Studio, you must select it for the source control system to work with Tango Editor. For example, in Microsoft's SourceSafe 5.0 installer, the option is **Enable SourceSafe Integration**.

You can perform common source control operations for managing your files, such as getting the latest versions, checking in and checking out, and adding to and removing from source control. You can also refresh your source control system's database and launch your source control user interface from within Tango Editor.

The **Source Control** menu and commands appear as follows.



**Note**

- Source control commands only appear in Tango Editor if you have a source control system installed on your machine. You must have your source control system's client software installed on the same machine as Tango Editor.
- The commands appearing in the **Source Control** menu depend on the source control system you are using. Commands vary from system to system. The commands appearing in this document are examples of the commands you may see.

Source control works only when the project is under source control. If it is not, all source control commands are disabled, except for **Add to Source Control**.

## Adding Files to Source Control

You must have your Tango project under source control before you can add individual files to source control. If your Tango project does not already exist, first create it as described in "Creating a New Project" on page 61.

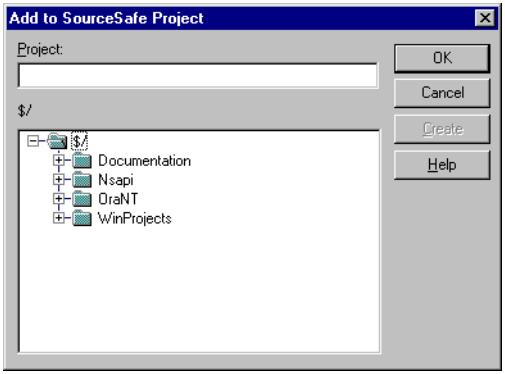
**Note** A *project* is a file denoted by the ".tep" extension. You add a project to source control the same way you add a file.

### *To add files to source control*

1. In the Project Workspace, select the files you want to add to source control.
2. Do either of the following:
   - From the **Project** menu, choose **Source Control**, then **Add to Source Control**.
   - Right click the selected files, and choose **Add to Source Control** from the context sensitive menu.

   If your source control system is not currently active, you are asked to log in first. Your source control system's login dialog box appears so you can log in as you would normally.

The **Add to {Source Control System} Project** dialog box for your particular source control system appears.




**Note** Depending on which source control system you are using, this dialog box may look different.

**3** Do either of the following:

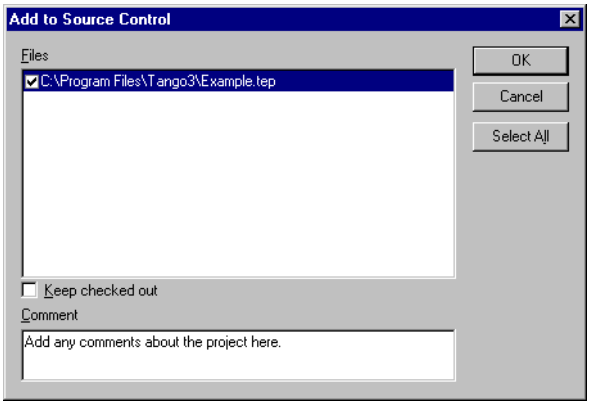
- Select from your source control database the project you want to save the selected files in.
- Enter a new project name, select its location in the database, and click **Create**.

  The new project is created in the location specified.

**4** Click **OK** to add the files to source control.

**Note** If you click **OK** without first creating the project in source control, Tango asks if you want the named project created for you. Click **Yes** to create the project or **No** to cancel.

The Add to Source Control dialog box appears.




The selected files and working folders appear in the **Files** list.


If you want to add the files to source control and check them out at the same time, enable the **Keep checked out** option. Otherwise, the files are added to source control but are not checked out.

5 Click **OK**.

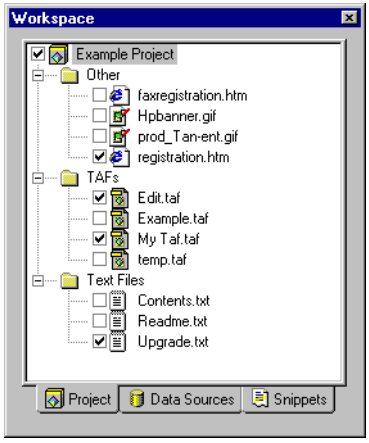
The Project Workspace changes to show checkboxes beside the file names. The checkboxes provide a convenient means of seeing the source control state of the project file and the files it contains.

The following table shows the various states a file can be in, as indicated by its checkbox, and what each state means.

| Checkbox | State of File |
|---|---|
| ☐ | Under source control and available for checking out. |
| ☑ | Under source control and checked out. |
| ▢ | Not under source control, but a member of a project under source control. |
| 🔒 | Under source control, but already checked out by another user. |

Remember you must have your Tango Editor project under source control before you can add individual files to source control.

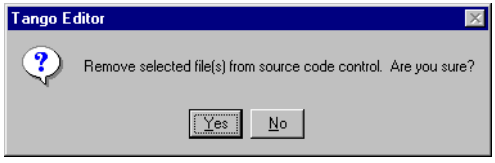
The following is an example of a project and files under source control.




## Removing Files From Source Control

***To remove files from source control***

1. In the Project Workspace, select the files you want to remove from source control.
2. From the **Project** menu, choose **Source Control**, then **Remove from Source Control**.
3. When asked if you want to remove the selected files from source control, click **Yes** to remove the files or **No** to cancel.




## Opening a Tango Project Already Under Source Control

When you open a Tango project already under source control, Tango Editor automatically gives you access to your source control system's functionality.

If your source control system is active when you open the Tango project, Tango Editor's source control features are made active. If it is not, your source control system's login dialog box appears so you can log in as you would normally.

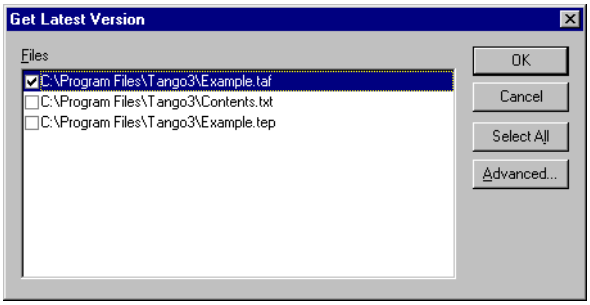
## Getting the Latest Version of Files

When you use the **Get Latest Version** command, Tango Editor allows you to view, but not modify, files. This command copies the files from the current source control project into your working folder. The files retrieved are read-only so modifications cannot be saved.

### *To get the latest version of files*

1. In the Project Workspace, select the files you want to get the latest version of. If you select a folder, all the files in the folder are automatically selected.
2. From the **Project** menu, choose **Source Control**, then **Get Latest Version**.

   The Get Latest Version dialog box appears, listing the files you can perform the **Get Latest Version** operation on.



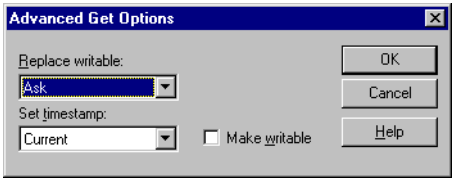

   The **Files** list shows the files currently in the Tango project. The files you selected in the Project Workspace are already selected to perform the **Get Latest Version** operation on.

   To indicate which files you want to get the latest version of, if different than those shown, select (☑) or deselect (☐) the corresponding file's checkbox.

   To select all the available files, click **Select All**.

3. If you want to set advanced options for **Get Latest Version**, click **Advanced**.

   An Advanced Get Options dialog box appears.



Because the options appearing correspond to the options for your particular source control system's get latest version feature, they may appear differently than the example shows. Click **Help** for a description of each of the available advanced options and how to set them.

Once set, click **OK** to return to the Get Latest Version dialog box.

4 Click **OK** to get the latest version of the selected files.

## Checking Out Files

When you use the **Check Out** command, Tango Editor copies the latest version of the selected files from the current source control project into your current working folder, and locks the source control system master copy.

### *To check out files under source control*

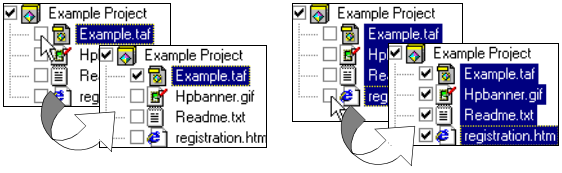
See also "Checking In Files" on page 75.

1 In the Project Workspace, select the files you want to check out. If you select a folder, all the files in the folder are automatically selected.

2 Do one of the following:

- Click the checkbox.

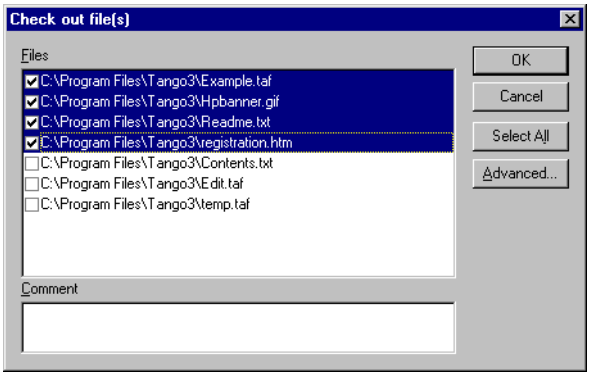
  If you select an individual file, click its checkbox. If you select multiple files, click any file's checkbox to check out all the selected files.




- Right click the selected files, and choose **Check Out** from the context sensitive menu.
- From the **Project** menu, choose **Source Control**, then **Check Out**.

For more information on setting source control preferences, see "Source Control" on page 309.

The selected files are automatically checked out (display of the Check Out Files dialog box is suppressed). If you enable the **Use dialog for checkout** option in Tango Editor's source control preferences, the Check Out File(s) dialog box appears.




The **Files** list shows all the files available for checking out. The files you selected in the Project Workspace are already selected to perform the **Check Out** operation on.

**Note** If you enable the **Use only selected files in dialogs** in Tango Editor's source control preferences, this dialog box lists only the files you selected in the Project Workspace for checking out.

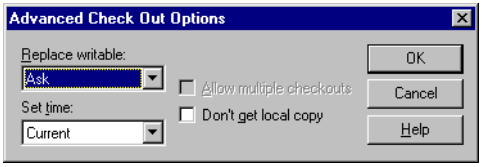
3 To indicate which files you want to check out, if different than those shown, select (☑) or deselect (☐) the corresponding file's checkbox.

To select all the available files, click **Select All**.

4 In the **Comment** area, you can add any comment, such as a brief description of the reason for the check out.

5 If you want to set advanced check out options, click **Advanced**.

The Advanced Check Out Options dialog box appears.



Because the options appearing correspond to the options for your particular source control system's check out feature, they may appear differently than this example shows. Click **Help** for a description of each of the available advanced options and how to set them.

Once set, click **OK** to return to the Check Out File(s) dialog box.

6 Click **OK** to check out the selected files.

## Checking In Files

When you use the **Check In** command, Tango Editor updates your source control system with changes made to the checked out file, and unlocks the source control system master copy.

### *To check in files under source control*

See also "Checking Out Files" on page 73.

1 In the Project Workspace, select the files you want to check in. If you select a folder, all the files in the folder are automatically selected.

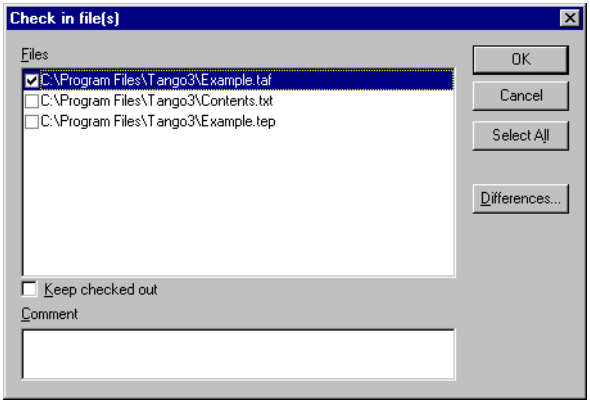
2 Do one of the following:

- Click the checkbox.

  If you selected an individual file, click its checkbox. If you selected multiple files, click any file's checkbox to check in all the selected files.

- Right click the selected files, and choose **Check In** from the context sensitive menu.
- From the **Project** menu, choose **Source Control**, then **Check In**.

The Check In File(s) dialog box appears.



The **Files** list shows all the files available for checking in. The files you selected in the Project Workspace are already selected to perform the **Check In** operation on.

**Note** If you enabled the **Use only selected files in dialogs** in Tango Editor's source control preferences, this dialog box lists only the files you selected in the Project Workspace for checking in.

For more information on setting source control preferences, see "Source Control" on page 309.

3 To indicate which file(s) you want to check in, if different than those shown, select (☑) or deselect (☐) the corresponding file's checkbox.

To select all the checked out files, click **Select All**.

4 In the **Comment** area, you can add any comment, such as a brief description of the reason for the check in.

5 To see the differences between the working folder version of the selected file and the source control database version of the file, click **Differences**.

Because the **Differences** feature depends on your particular source control system, you should refer to your source control user documentation for a description of the visual difference feature and how to use it.

When you exit the differences feature, the Check In File(s) dialog box reappears.

6 Click **OK** to check in the selected files.

## Undoing Checked Out Files

When you use the **Undo Check Out** command, Tango Editor cancels the check out operation, undoing all changes. In other words, you lose any changes you made to the working copies of your files. You must have a working folder set for the undo operation to work properly.

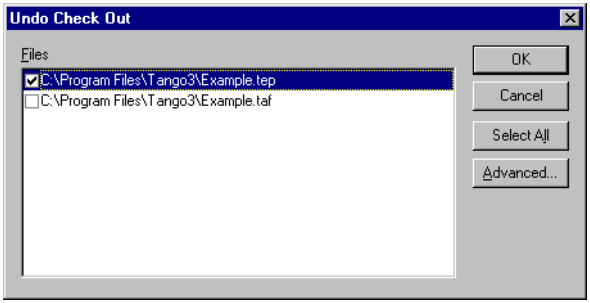
### *To undo checked out files*

1 In the Project Workspace, select the checked out files you want to undo. If you select a folder, all the files in the folder are automatically selected.

**2** Do either of the following:

- Right click the selected files, and choose **Undo Check Out** from the context sensitive menu.
- From the **Project** menu, choose **Source Control**, then **Undo Check Out**.

The Undo Check Out dialog box appears, listing the files you can perform the undo check out operation on.




The **Files** list shows all the files currently checked out. The files you selected in the Project Workspace are already selected to perform the **Undo Check Out** operation on.

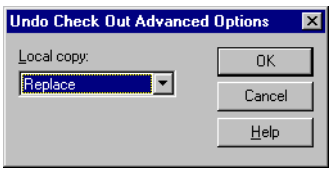
**Note** If you enabled the **Use only selected files in dialogs** in Tango Editor's source control preferences, this dialog box lists only the files you selected in the Project Workspace for performing the **Undo Check Out** operation on.

To indicate which checked out files you want to undo, if different than those shown, select (☑) or deselect (☐) the corresponding file's checkbox.

To select all the checked out files, click **Select All**.

**3** If you want to set advanced undo check out options, click **Advanced**.

An Undo Check Out Advanced Options dialog box appears.



Because the options appearing correspond to the options for your particular source control system's undo check out feature, they may appear differently than the example shows. Click **Help** for a description of each of the available advanced options and how to set them.

Once set, click **OK** to return to the Undo Check Out dialog box.

4 Click **OK** to undo the check out of the selected files.

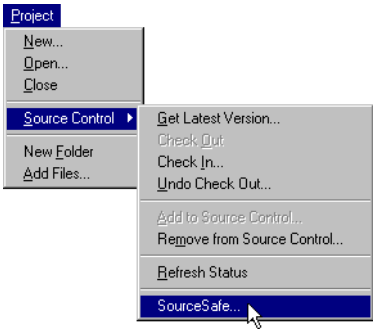
## Refreshing File Status

When you want to update the status of the files currently under source control, use the **Refresh Status** command. This command updates the check in and check out status of all files under source control.

## Launching Your Source Control System

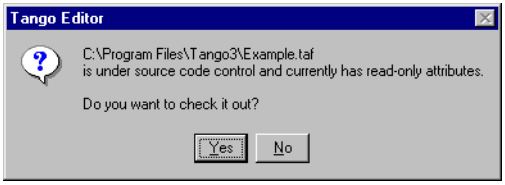
You can launch your source control system at any time from Tango Editor.

From the **Project** menu, select **Source Control**, and the name of your particular source control system, which appears at the bottom of the menu.



## Opening a File Under Source Control

If you try to open a file that is currently under source control and not checked out, Tango Editor prompts you to check it out first.




Remember files not checked out are read-only. You must check out a file before any changes you make can be saved.

Click **Yes** to check out the application file or **No** to open it as a read-only file.

The Check Out File(s) dialog box appears.

For more information, see “Checking Out Files” on page 73.

CHAPTER FOUR

# Using Data Sources

4

*Data Source Basics, Operations, and Properties*

A Tango *data source* contains all the information needed to connect to a particular database. You use data sources to tell your Tango applications which databases to connect to. You use Tango Editor to create and manage data sources.

Both Tango Editor and Tango Server need to see the data source required to access a particular database. Tango Editor uses a data source—via the Data Source workspace—so you can see information in the form of table and columns. Tango Server requires the same data source so it can access the database tables and columns specified within the application file.

The data source *properties* show the information about the data source, including information about its tables and columns.

This chapter covers the following topics:

- understanding Tango data sources
- the Data Sources workspace
- using data sources, including creating, modifying, and deleting them
- working with data source properties
- connecting to data sources
- assigning data sources to actions.

# About Data Sources

Tango supports two types of data sources:

Tango Enterprise for Mac OS supports two other data source types: FileMaker Pro and Butler SQL (DAM).

- **ODBC**, in conjunction with third-party drivers, supports connections to a wide variety of database types.
- **Oracle** supports connections to Oracle databases.

## ODBC Data Sources

Open Database Connectivity (ODBC) is a standard developed by Microsoft to allow applications like Tango to communicate with a wide variety of databases from different vendors. An ODBC client application talks to the ODBC driver manager that in turn talks to a database driver for a specific type of database.

An ODBC driver is a kind of translator. It converts the standard ODBC requests made by the application into a format that can be understood by the target database system. ODBC drivers are available for accessing many database management systems (DBMS). Microsoft Access, Excel, Oracle, SQL Server, Informix, Sybase, and Butler SQL are some examples of databases that may be accessed through ODBC.

Before creating an ODBC data source, you must set up your database server and create or install a database on this server. Depending on the database system, you may also need to install and configure additional software to allow you to connect to the server. Consult your database software and ODBC driver documentation for specific instructions. The ODBC system software is installed automatically with Tango, and you need an ODBC driver for the type of database you are connecting to.

For more information, consult the documentation that accompanied your ODBC drivers.

Some ODBC drivers are self-contained. The driver itself accesses and updates the database files. For these ODBC drivers, no other software is required.

**Note** If you are trying to connect to Butler SQL on a Windows computer, you must use ODBC.

## Oracle Data Sources

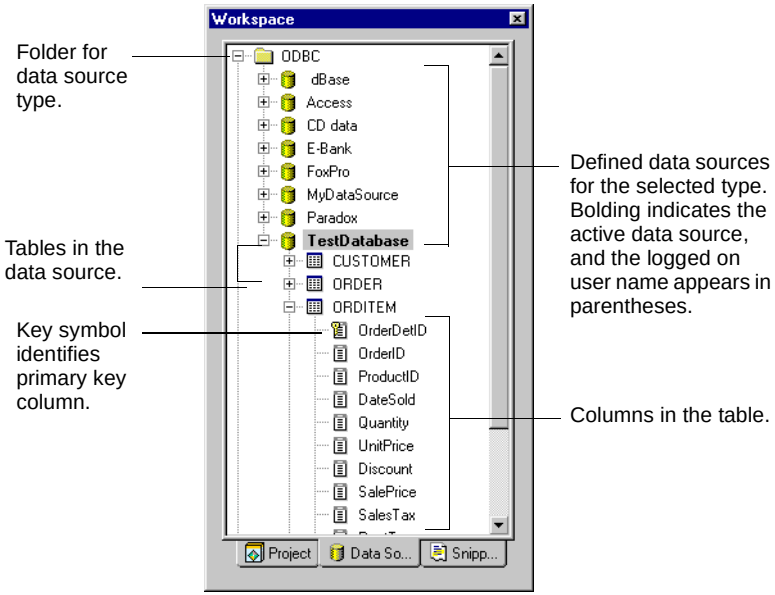
Oracle is a high-performance client/server DBMS. To create and use Oracle data sources, you must have Oracle's SQL*Net installed. Tango supports SQL*Net versions 7.1 and 7.3, and greater.

# The Data Sources Workspace

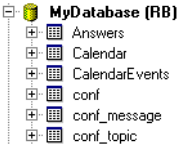
You perform most data source operations in the Data Sources workspace. To open the Data Sources workspace, click the Data Sources tab in the workspace window.

When you open the Data Sources workspace, you see a folder for each type of data source that Tango supports.

- Expanding a folder shows the defined data sources for that type.
- Expanding a data source show the tables in that data source. Depending on the settings for the data source, you may need to enter user name and password information before a connection can be made.
- Expanding a table shows the columns in that table.




The bolded data source name in the workspace indicates the currently active data source—that is, the data source assigned to the front-most open action window. If no database actions are active, no data source names are bolded.




Once a connection is made to a data source, the user name used for the connection appears in parentheses after the data source name. This avoids any confusion when different logins are being used for the same data source.

## Using the Primary Column Key

Primary Key

A *primary key* is a column (or combination of columns) whose value uniquely identifies each row in a table. For example, a customer number might be the primary key in a customers table. In the Data Sources workspace, Tango Editor displays the column / key icon next to a column name to show that it is part of the primary key for the table.

Tango builders rely on the primary key column values in various places to identify specific records. When using the builders, it is important to first check that the primary key for each table involved is set correctly. If the specified column or columns do not uniquely identify each record in a table, unexpected results can occur when executing the file. For example, if you mistakenly set the primary key for a customer table to the “state” column (many customers likely share the same state), using the resulting file to delete a particular customer deletes *all* the customers in the same state.

When connecting to a data source, Tango Editor chooses default primary keys by scanning each table for the first column with an appropriate data type (numeric or character).

Primary key columns appear with a column / key icon to differentiate them. You identify a column as a primary key or not by doing one of the following:

- Right click the column to display a menu of commands, and choose **Primary Key Column**. A checkmark beside the command indicates a primary key column.
- From the Data Source menu, select **Primary Key Column** to show a checkmark or not.
- Open the column properties window, and enable or disable the **Primary Key** option.

# Data Source Operations

## Creating a Data Source

### ODBC

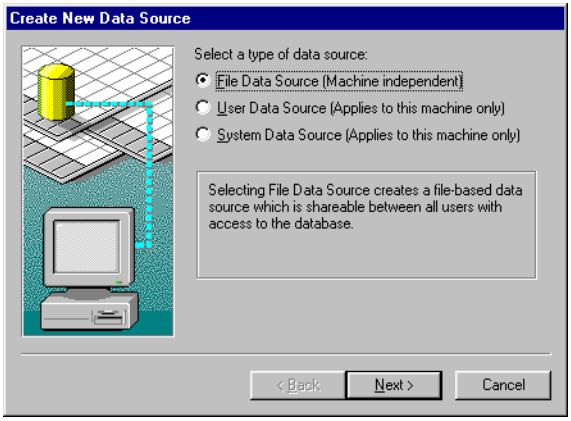
Before you create an ODBC data source, make certain the required ODBC drivers are installed and your database server is running (or, for drivers that access local files, your database files are available).

#### *To create an ODBC data source*

You can also right click the Data Sources workspace to display a menu to select the **New** command from.

1 From the **Data Source** menu, click **New** then click **ODBC** from the submenu.

The Create New Data Source dialog box appears.




**Note** The dialog box appearing may look different. The appearance varies depending on the version of ODBC you have.

2 Select **System Data Source** from the list of data sources.

**Note** Tango only supports System data sources.

3 Choose **Next**.

See your ODBC driver documentation for detailed configuration instructions.

The Create New Data Source dialog box guides you through the rest of the process. Follow the instructions that appear.

### Oracle

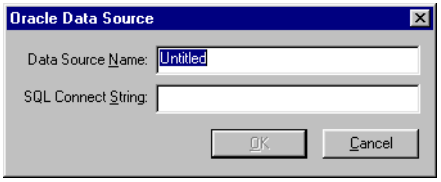
Before creating an Oracle data source, make sure the correct Oracle client software is installed and the database server you want to connect to is available on the network.

#### *To create an Oracle data source*

You can also right click the Data Sources workspace and choose **New** from the context-sensitive menu.

1. From the **Data Source** menu, click **New** then click **Oracle** from the submenu.

   The Oracle Data Source dialog box appears.




2. In the **Data Source Name** field, type a name for the data source.
3. Type the SQL Connect string in the field provided.

   With current versions of SQL*Net, this should be the name of a database alias set up on your computer with the Oracle EasyConfig application.

**Note** Tango only supports SQL*Net versions 7.1 and 7.3, and greater.

4. Choose **OK**.

   The new data source is added to the Data Sources workspace.

## Modifying a Data Source

You can also right click the Data Sources workspace to display a menu to choose **Modify** from.

For information on changing the settings for each type, see "Creating a Data Source" on page 85.

#### *To modify a data source*

1. Select the data source you want to edit.
2. From the **Data Sources** menu, choose **Modify**.

   The data source modification dialog box appears.
3. Modify the desired settings for the data source.

If a modified data source is already loaded, the data source is reloaded automatically using the new settings.

For Oracle data sources, modifying a data source does *not* affect Tango application files already using that data source, even if they are open when the modification is made. To update a document

with the new settings, you must reassign the data source to the document by choosing **Set Data Source** from the **Data Source** menu.

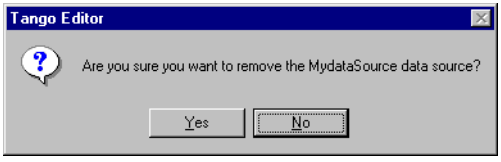
## Deleting a Data Source

### *To delete a data source*

1. Select the data source you want to delete.
2. Do one of the following:
   - From the **Data Sources** menu, choose **Delete**.
   - From the **Edit** menu, choose **Delete**.
   - Right click the Data Sources workspace, and choose **Delete**.
   - From the main toolbar, click the Delete icon.
   - On the keyboard, press DELETE.

   A confirmation dialog box appears.

To suppress the confirmation dialog box, hold down CTRL when deleting.




3. Choose **Yes**.

   The data source is deleted.

## Reloading a Data Source

If the structure of your database changes while Tango Editor is open, you need to reload the data source.

### *To reload a data source*

1. Select the data source you want to reload.
2. From the **Data Sources** menu, choose **Reload**.

You can also right click the Data Sources workspace to display a menu to choose **Reload** from.

The log in information is as specified in the data source's Log In properties.

## Handling Unknown Data Sources

For Oracle data sources, actions do not rely on anything in Tango Editor's list of data sources to connect to a data source. When connecting, Tango Editor uses only the information stored in the data source. For this reason, Tango Editor could connect to an action's data source while not having a data source defined with

matching parameters. Such a data source is called an *unknown* data source.

**Note** ODBC data sources are an exception. Tango ODBC data sources and those defined on the computer are the same.

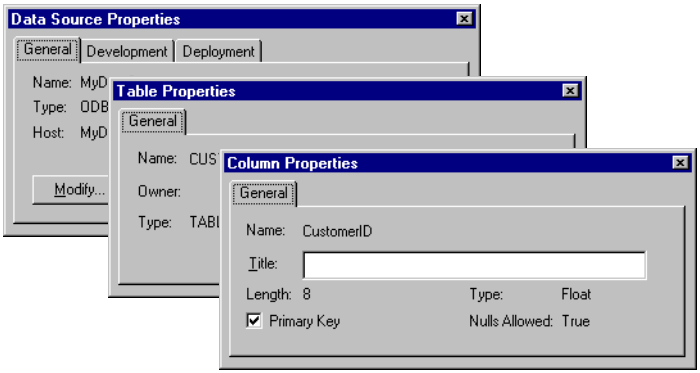
When you open an action having an undefined data source in Tango Editor, yet Tango Editor is able to connect to it, the unknown data source is added automatically to the list in the Data Sources workspace. The log in information for the new data source is set to the defaults.

For more information on the default log in settings, see "Development Tab" on page 90.

This happens even if there is a data source defined with the same parameters, but with a different data source name. That is, all of the pieces of data source information must match in order for an existing one to be used.

# Working With Data Source Properties

When you select a data source, a table, or a column in the Data Sources workspace, the corresponding Properties window appears. These windows allow you to view information about the selected object.

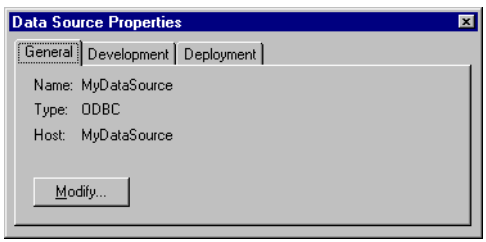



## Setting Log In Options for a Data Source

The Data Source Properties window contains three tabs: General, Development, and Deployment. Click a tab to display the corresponding properties section.

### General Tab

The General section shows the information making up the data source. The following is an example of the information for an ODBC data source called “MyDataSource”.



The data source name and type appear for all data sources, but the information in the other fields depends on the type of data source.

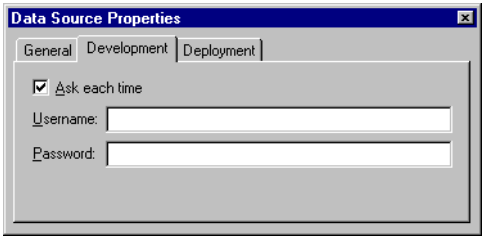
| Data Source Type | Other Information |
|---|---|
| ODBC | [None] |
| Oracle | Connection string |

To edit the selected data source, choose **Modify**.

The data source editing dialog box for the data source appears. When you close the dialog, any new settings appear in the General section.

## Development Tab

The Development section shows the log in information required by Tango Editor for connection to the data source. For example, the ODBC version looks as follows.




The Development tab in the properties dialog box asks for a user name and password for each type of data source.

The **Ask each time** option means Tango Editor always asks for connection information when needed. For new data sources, this option is selected by default.

## Deployment Tab

The Deployment section allows you to specify different log in information to be used when Tango Server executes the action the data source is assigned to. The **Username** and **Password** fields may also contain meta tags that are substituted when Tango Server executes the application file.

The **Same as development** option is the default, meaning the user name and password used during development is also used by Tango Server.

When you right click the **Username** and **Password** fields, a menu appears containing standard editing commands, as well as the **Insert Meta Tag** command. You can use **Insert Meta Tag** to insert many of the commonly used Tango meta tags.

Before connecting to a data source, Tango checks the data source parameters for the presence of meta tags. If meta tags are found, the substitution is performed, and the results are used for establishing the connection. If no meta tags are found, the data source parameters are passed literally.

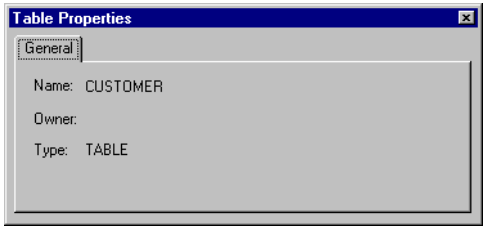
The following example shows that user name and host name are obtained from the user variables *username* and *hostname*, respectively. The user password is taken from the file, whose name corresponds to the user name, anchored by the `.pwd` extension.

Username: `<@VAR NAME="`*username*`">`

Password: `<@INCLUDE FILE="<@VAR `*username*`>.pwd">`

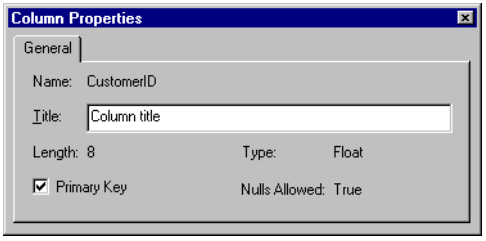
## Table Properties

Table Properties shows the name, owner, and type of table. For example:



## Column Properties

The Column Properties window displays the name, title, data type, length, whether nulls are allowed or not, and whether the column is a primary key or not. For example,




In this dialog box, you can edit the **Title** field, and select the **Primary Key** option.

**Note** This title is used by the builders as the default HTML display title for the column. See “Column Options” on page 156.

# Connecting to Data Sources

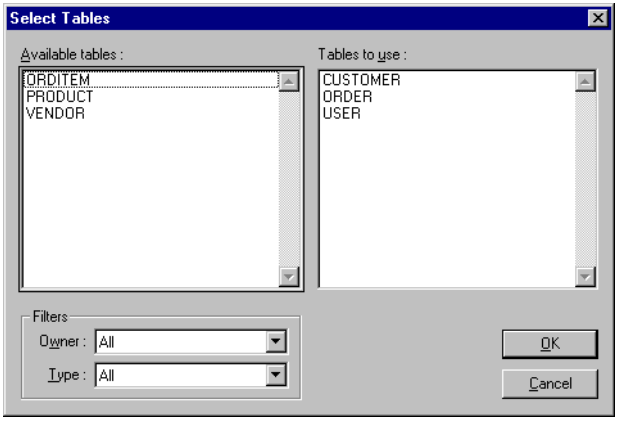
When you expand a data source in the Data Sources workspace you have not connected to, the log in information specified in the Data Source Properties Development window is used for the connection. If you selected the **Ask each time** option, the Log In dialog box appears allowing you to type your user name and password.

For more information, see “Connecting to Large Data Sources” on page 93.

If a data source has more than the specified minimum number of tables (the default is 25), the Select Tables dialog box appears, allowing you to work with a more manageable subset.

## Connecting to Large Data Sources

When Tango Editor connects to a data source containing more than 25 tables, it presents the Select Tables dialog box, allowing you to select which tables you want to work with.




The **Available tables** list shows the tables in the data source. Drag the tables you want to work with from this list into the **Tables to use** list. You can use the **Owner** and **Type** drop-down lists to filter the tables shown in the **Available tables** list.

For example, to show only tables owned by a specific user, select that user from the **Owner** drop-down list. To show only system tables, select **SYSTEM TABLE** from the **Type** drop-down list. (The contents of these drop-down lists are determined by the data source; only owners and types existing in the database are listed.)

You can also invoke the Select Tables dialog box at any time to select the tables to work with. From the Data Source menu, choose **Select Tables** to invoke the dialog box.

## Editing and Executing Files on Different Computers

When connecting to a data source, Tango relies on configuration information not included in the Tango application file itself. This becomes an issue when Tango Editor and Tango Server reside on different computers, and when editing a file created on a different computer. Tango cannot connect to the data source unless the computer is set up correctly.

The following explains what pieces of data source information are stored in the file, what ones are not, and how to ensure a file works on a computer other than the one it was created on.

### ODBC Data Sources

Files assigned ODBC data sources have these pieces of information stored in them:

- ODBC data source name
- user name
- password

For the data source connection to be made on another computer, a data source with the same name pointing to the original database must exist. The user name and password must also be valid for the server pointed to by the data source.

### Oracle Data Sources

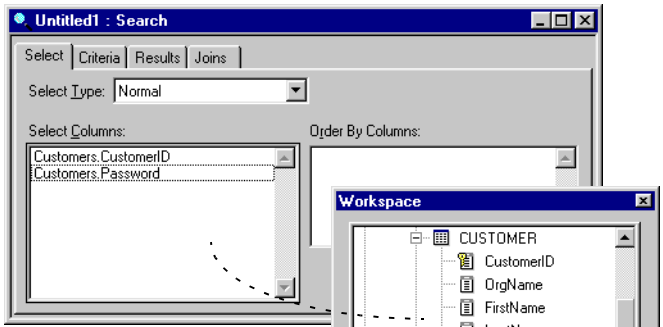
Files assigned Oracle data sources have these pieces of information stored in them:

- SQL connect string or database alias name
- user name
- password

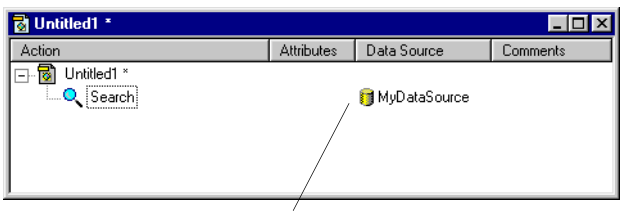
If you specified a SQL connect string (such as `T:199.230.9.8:ORCL`) when defining the data source, your Tango file works on any computer the string points to that has access to the Oracle database server. This is because all the connection information is stored right in the string.

# Assigning Data Sources to Actions

To assign a data source to an action, simply drag a column from a data source to an action editing window.




Dragging a column into an action editing window assigns the data source to the action.




The data source icon and data source name appear next to the assigned action.

**Note** Some actions (for example, Transaction actions) do not have columns. If you drag a database action that has no columns you are prompted to select a data source.

For more information, see “Setting Log In Options for a Data Source” on page 89.

If Tango Editor has not yet connected to the data source, a log in dialog box may appear. This dialog box only appears if you have the **Ask each time** option enabled, which is the default, in the Development section of the Data Source Properties window.

If an action already has a data source assigned to it and you drag a column into it from a different data source, you are asked if you want to cancel the operation or to use the new data source instead.

**Note** If there are differences in the structures of the databases, changing an action's data source may cause DBMS errors when the action is executed.

If you use a new data source, Tango Editor scans the affected actions to update the table owner information to match the new data source.

CHAPTER FIVE

# *Using Snippets*

5

## *Snippets Basics and Operations*

*Snippets* are named pieces of text, such as meta tags, HTML tags, standard headers and footers, plain text, JavaScript, and SQL. Snippets are a good way of saving text, HTML markup, or other commands that you use frequently. You can insert snippets into most text fields throughout Tango Editor.

Tango Editor comes with a large defined set of snippets and also lets you create your own snippets.

This chapter discusses how to:

- use snippets
- create new snippets.

# About Snippets

Snippets are used to quickly access text, meta tags, and HTML that you use often. You use snippets by dragging them into an editing window or double clicking on a snippet when you have an HTML editing window open.

A special feature of snippets lets you *surround* a selection of text with tags or meta tags, in addition to putting text, tags, or meta tags at a particular place.

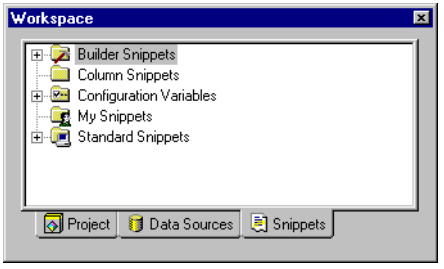
Your snippets are saved as text files within the `My Snippets` folder inside the `Windows/`*`your_login`* folder (under Windows 95) or inside the `Winnt/Profiles/`*`your_login`* folder (under Windows NT). You can edit Tango snippet files with a text editor or HTML editor, or edit them within the Snippets workspace.

## The Snippets Workspace

You manipulate and manage snippets in the Snippets Workspace.

### *To display the Snippets Workspace*

Open the Workspace window by choosing **Workspace** from the **View** menu and click the Snippets tab.




You expand a folder by clicking the "+" sign to view its contents. Folders and snippets appear alphabetically.

Snippets are grouped in five folders:

For more information, see “Column Snippets” on page 106.

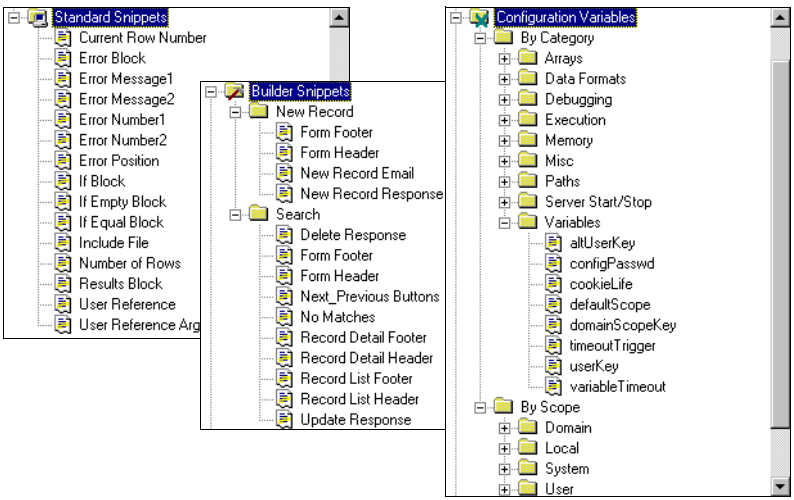
- **Builder Snippets** contains pre-installed snippets used by the Search and New Record builders. These may be edited to change default HTML used in various places in the builders.
- **Column Snippets** contains context-sensitive column snippets, used in the Search action, the Search Builder, and the New Record Builder.

For more information, see “Shortcuts to Configuration Variable Assignments: Snippets” on page 137.

- **Configuration Variables** contains pre-installed snippets of the Tango configuration variables. They are organized by category and by variable scope.
- **My Snippets** contains snippets you create and edit.
- **Standard Snippets** contains pre-installed snippets, which are not editable.

For more information on configuration variables, see the *Meta Tags and Configuration Variables* manual.

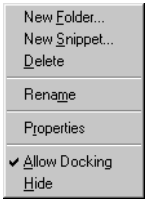
The following are examples of some of the snippet folders and snippets available in the Snippets Workspace:




## Context-Sensitive Menus in the Snippets Workspace

You can access a variety of commands when you right click items in the Snippets Workspace. There are two different context sensitive menus: one for folders, and one for snippets. In both menus, the Workspace commands—**Allow Docking** and **Hide—**are always active.

- When you right click a snippet folder, the following context-sensitive menu appears:

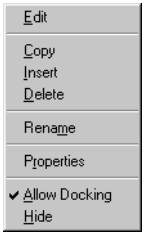



  If you are clicking on the My Snippets folder, you can create a new folder or new snippet, or show the properties of the My Snippets folder; the **Delete** and **Rename** commands are inactive.

  If you are clicking on a folder within the My Snippets folder, you can create a new folder or new snippet from this menu, delete an existing folder, rename it, or show its properties.

  If you are clicking on a non-editable folder, you can only show its properties.

- When you right click a snippet, the following context-sensitive menu appears:




  The **Copy** command copies the content of a snippet to the Windows clipboard. When you have a window open where you can insert a snippet (for example, an HTML editing window), the **Insert** command is active.

  If you are clicking on a non-editable snippet, you cannot delete, edit, or rename the snippet, but you can copy its content to the clipboard or or show its properties.

  If you are clicking on an editable snippet (within the My Snippets folder), you can edit, copy, insert, delete, and rename snippets, as well as show their properties.

# Working With Snippets

## Inserting Snippets

### *To insert a snippet into an application file*

Do one of the following:

- Drag the snippet to the desired text field (for example, the Results editing window).
- Place your cursor where you want the snippet inserted and double click the snippet (HTML editing windows only).
- Right click on the snippet and choose **Insert** from the context sensitive menu that appears.

**Tip** The content of a snippet appears as a Windows tool tip if you place your mouse cursor over the snippet.

You can select and drag several snippets at the same time.

### *To drag multiple snippets*

1. Select the snippets you want to drag by doing either of the following:
   - Clicking a snippet, holding down the SHIFT key, and clicking to select all the items in a list.
   - Clicking a snippet, holding down the CTRL key, and selecting additional snippets (discontiguous selection).
2. Drag the snippets into the desired text field.

You can also double click and have the multiple snippets inserted at the insertion point, or right click on the snippets you have selected and choose **Insert** from the context sensitive menu that appears.

## Creating Snippets

### *To create a snippet with existing text*

Drag text to a folder in the Snippets workspace.

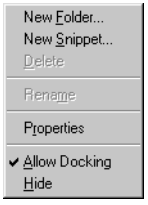
A snippet is created containing the text you dragged. The name of the new snippet defaults to “Untitled” and is editable. Type in a new name.

### *To create a snippet with new text*

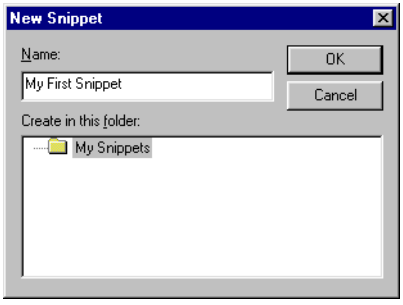
**1** Do either of the following:

- From the **Edit** menu, click **Snippet,** then click **New** from the submenu.


- Right click in the My Snippets folder of the Snippets Workspace and select **New Snippet** from the context-sensitive menu that appears.




The New Snippet window appears.



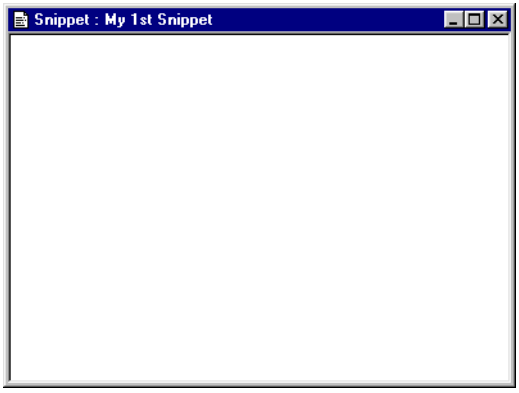
2 Fill in the window as follows.

**Name.** Enter the name you are assigning to the new snippet.

**Create in this folder**. The **My Snippets** folder and any subfolders are shown by default. Select where you want to create the new snippet.

3 Click **OK** when you have given the new snippet a name and location.

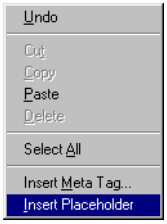
The Snippet contents window appears.




4 Type in the content of your snippet or drag in text from elsewhere.

## Using Placeholders in Snippets

When you right click the Snippet contents window, the **Insert Placeholder** command is listed in the context-sensitive menu that appears, in addition to the standard Tango editing commands.




**Insert Placeholder** inserts a special symbol—the yen symbol, ¥—into the Snippet contents window. This placeholder character allows you to create snippets that put HTML or meta tags around text you select before inserting the snippet. For example, you can create a snippet of the following text:

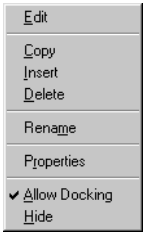
```
<H1>¥</H1>
```

To use this snippet while editing HTML, you would select the text you want to apply the <H1> format to, and then drag the snippet into the editing window or double click the snippet. The selected text is surrounded by the <H1> and </H1> tags.

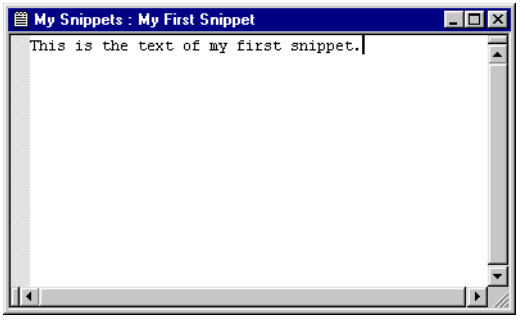
## Editing Snippets

### *To edit the contents of a snippet*

1 Right click the snippet you want to edit and choose **Edit** from the context-sensitive menu that appears.




The Snippet content window appears.




2 Edit the text.

You can edit multiple snippets. Choosing **Edit** from the right click menu opens the contents window for all the selected snippets.

## Creating a Snippets Folder

You can create new folders for holding your snippets in the **My Snippets** folder. This feature is useful if you want to categorize your snippets by type (for example, meta tags, SQL, HTML, and text snippets).

### *To create a new folder in the My Snippets folder*

1 Right click the My Snippets folder, and choose **New Folder** from the context sensitive menu that appears.

A folder called **Untitled** appears. The name is already selected so you can change it.

2 Type in a different name.

## Renaming Snippets and Snippet Folders

### *To change the name of a snippet or snippet folder*

Do one of the following:

- Click a snippet or snippet folder and click again.
- Right click the snippet or snippet folder, and choose **Rename** from the context-sensitive menu that appears.

The name of the snippet or snippet folder becomes editable; type in the new name.


**Note** Snippet names must be unique inside a folder.

## Copying, Moving, and Deleting Snippets

### *To copy a snippet to the My Snippets folder for editing*

Drag the snippet from the **Standard Snippets**, **Configuration Variables** or **Column Snippets** folders to the **My Snippets** folder.

You can move snippets to a different folder within the **My Snippets** folder by dragging snippets to their new location.

### *To duplicate a snippet within the My Snippets folder*

Hold down the CTRL key, and drag the snippet.

### *To delete a snippet or snippet folder*

Click the snippet or snippet folder, and do one of the following:

- Right click the snippet or snippet folder, and choose **Delete** from the context-sensitive menu that appears.

- On the main toolbar, click the Delete icon.
- From the **Edit** menu, choose **Delete**.
- Press the DELETE key.

A dialog box asking you if you want to delete the snippet appears. Click **OK**.

**Note** You can delete items from only the **My Snippets** and **Builder Snippets** folders.

You can delete multiple snippets. Choosing **Delete** from the right click menu deletes the selected snippets.

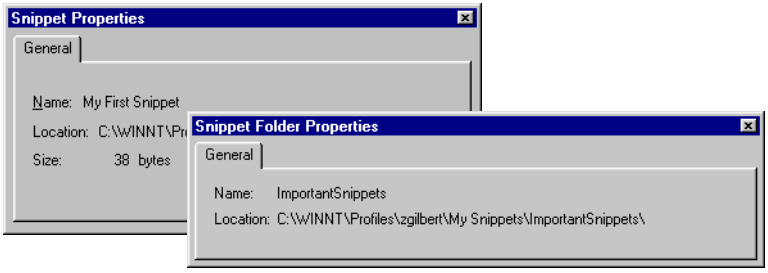
## Snippet Properties

***To see the properties of a snippet or snippet folder***

Select the snippet or snippet folder, and do one of the following

- From the **View** menu, choose **Properties**.
- Type ALT-ENTER.
- Right click the snippet or snippet folder and choose **Properties** from the context-sensitive menu that appears.

The Snippet Properties or Snippet Folder Properties dialog box appears:




These dialog boxes list the name and location of the snippet or snippet folder, and the size of the snippet.

## Column Snippets

The **Column Snippets** area of the Snippets workspace becomes active when the following are displayed:

- Results HTML of the Search action.

  The Search action's **Select columns** appear (and the **Snippets** tab is switched to and the **Columns Snippets** folder expanded, if necessary). (The Search action appears in the actions generated by the New Record Builder and Search Builder, as well as a separate action.)

- New Record Response HTML in the New Record Builder.

  Shows all the columns from the table being inserted into.

For more information, "<@COLUMN>" on page 60 of the *Meta Tags and Configuration Variables* manual.

The content of each snippet is `<@COLUMN name>`, where the column name is the name from the Search action or New Record Builder you are editing.

CHAPTER SIX

# *Using Meta Tags*

6

## *Understanding Meta Tags and How to Insert Them*

Meta tags are special tags that are entered in Tango Editor and are interpreted by Tango Server when your application files are executed. They can do many different things in an application file, from controlling the flow of information to performing an action on a data source to setting or retrieving variable values.

Meta tags look like HTML tags, with a starting and ending angle bracket ("<" and ">"), but the character after the starting angle bracket is "@", or in the case of a closing meta tag, "/@". When you are creating HTML in Tango Editor, you insert meta tags just like HTML tags. The user's Web browser never sees the meta tags, as they are interpreted by Tango Server before being sent to the Web browser.

You can find a complete and detailed list of Tango meta tags and their attributes in the *Meta Tags and Configuration Variables* manual.

This chapter covers the following topics:

- introduction to meta tags
- how to insert meta tags in Tango application files and HTML documents created with Tango.

# About Meta Tags

Meta tags are special commands to Tango that can do many things, including control execution of Tango application files, return values from a database, create variables, and return the values of variables. One of the places these tags have their effect is in the HTML returned by Tango Server to your Web browser; for example, Tango may return HTML to the browser using meta tags that refer to form field values (<@POSTARG>) and values returned from a database (<@COLUMN>).

Meta tags are interpreted by Tango Server at the application file execution time and the resulting values, if any, are substituted where the meta tags appear.

For detailed information on these and other meta tags, see the *Meta Tags and Configuration Variables* manual.

Most meta tags return values when interpreted by Tango Server. A few Tango meta tags that control the flow of information or assign values to variables do not return values. These include <@ROWS>, <@IF>, <@IFEQUAL>, <@IFEMPTY>, and <@ASSIGN> meta tags.

Meta tags begin with the "at" symbol, "@", to distinguish them from HTML tags. Closing meta tags begin with "/@". This documentation shows meta tags in uppercase, but meta tags are case insensitive; that is, <@if>, <@IF>, and <@iF> are all treated the same.

Meta tags often have attributes, much like HTML tags. These name / value attribute pairs specify required and optional attributes of the meta tag. For example:

```
<@ASSIGN NAME="last_name" VALUE="Flintstone">
```

This example assigns the value "Flintstone" to the variable last_name. You can leave the name of an attribute off in certain cases: if the attribute is required and in its standard position; however, it is recommended that you use attribute names to avoid ambiguity.

## Where You Can Use Meta Tags

Most meta tags can be used in all places in application files where text or HTML can be inserted, including these application file locations:

- attribute HTML that is attached to an action, including:
  - Results HTML
  - Error HTML
  - No Results HTML
- actions in an application file, including:
  - parameters in Search, Update, and Delete actions
  - column values in Update and Insert actions
  - Maximum Matches and Start Match fields in Search actions
  - External action parameters
  - File action parameters
  - Assign actions (both name and value)
  - If action parameters
  - custom and column references used in database actions
  - SQL entered into the Direct DBMS action window
- HTML files included using the `<@INCLUDE>` meta tag
- most attributes for other meta tags.

Where you can insert meta tags, the right mouse menu shows **Insert Meta Tag**.

## Combining Meta Tags

When you use meta tags in action fields or in attributes of other meta tags, you can use multiple meta tags and mix literal values with meta tags. For example, in a column value field parameter for an Insert action, you could specify:

```
<@POSTARG NAME=prefix><@POSTARG NAME=suffix>
```

This indicates the concatenation of the `prefix` and `suffix` form fields.

To give a long distance code in a standard format that includes spaces and meta tags, the parameter would look something like the following:

```
+1 <@POSTARG NAME=area_code> <@POSTARG
NAME=phone_num>
```

## Quoting Attribute Values

Only attributes that have spaces in them need to be quoted, but it is never wrong to quote attributes. Either single or double quotes can be used.

For more information on the rules for quoting attributes in meta tags, see “Quoting Attributes” on page 8 of the *Meta Tags and Configuration Variables* Manual.

For example:

```
<@CALC EXPR=3+4 PRECISION="2">
<@CALC EXPR="3+4" PRECISION="2">
```

Both examples are correct, as is the single quote

```
<@POSTARG NAME='homer'>.
```

**Tip** For new users of Tango, the best method to adopt is quoting all attribute values.

# Inserting Meta Tags

For more information, see “Working With Snippets” on page 101.

Meta tags can always be entered by typing them or dragging a meta tag snippet into your application file. There is also a shortcut to inserting many common meta tags: the Insert Meta Tag dialog box. This dialog box does not contain all of the meta tags; some must be typed in or dragged in from the Snippets Workspace.

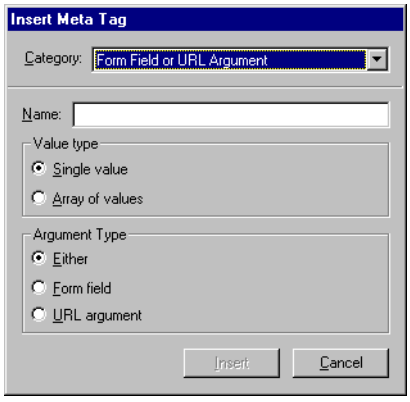
See the *Meta Tags and Configuration Variables* manual for a detailed list of meta tags and their options.

The **Insert Meta Tag** command, available from the context-sensitive right-mouse menu or the **Edit** menu, inserts a meta tag into an application file. This dialog box shows common meta tags in a category drop-down list. This provides a quick reference for many common meta tags.

### *To insert common meta tags into your application file*

1 From the **Edit** menu, choose **Insert Meta Tag**.

The Insert Meta Tag dialog box appears.




2 From the **Category** drop-down list, select one of the five options:

- Form Field or URL Argument
- Variable
- Current Date / Time
- Request Parameter
- Action Result Item

Selecting a category changes the dialog box to show the appropriate fields for the category of meta tag.

3 Select or enter the attributes necessary for the meta tag you are inserting.

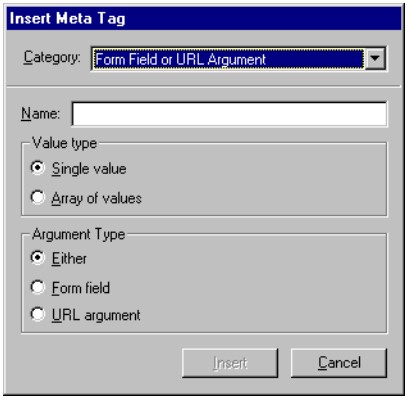
4 Click **Insert**.

The **Insert** button is enabled only when you have entered sufficient information to construct the meta tag.

The meta tag and its attributes are placed at the insertion point in your application file when **Insert** is clicked. The various types of meta tags you can insert are described in the following sections.

## Form Field or URL Argument

To insert a meta tag that returns the value of a form field or a URL argument, choose **Form Field or URL Argument** from the **Category** drop-down list. These meta tags are <@SEARCHARG>, <@POSTARG>, and <@ARG>. A name must be specified. (The name of an argument is assigned in the HTML form that is set up by the creator of a Web page.)




Once you have specified the name, the radio buttons have the following effects:

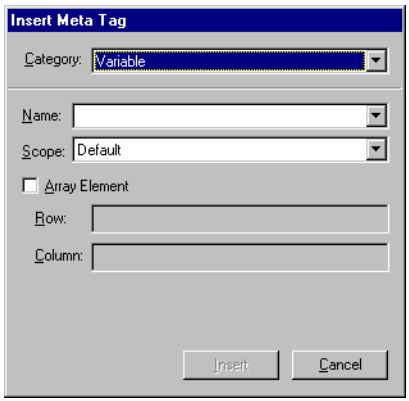
For more information, see "<@ARG>" on page 29, "<@POSTARG>" on page 111, and "<@SEARCHARG>" on page 127 of the *Meta Tags and Configuration Variables* manual.

- **Single value** has no effect on the inserted meta tag.
- **Array of values** adds the TYPE=ARRAY parameter to the meta tag.

  Use this option to get all values for form fields which may contain multiple values (such as lists).
- **Either** inserts <@ARG>.
- **Form field** inserts <@POSTARG>.
- **URL argument** inserts <@SEARCHARG>.

## Variables

To insert a meta tag that returns the value of a variable (the <@VAR> tag) with the Insert Meta Tag dialog box, choose **Variable** from the **Category** drop-down list.




For more information, see "<@VAR>" on page 154 of the *Meta Tags and Configuration Variables* manual.

The following attributes can be assigned for insertion of <@VAR> with the Insert Meta Tag dialog box:

- **Name** contains an alphabetized list of variables *assigned to* (via Assign actions) in the current application file. Select one of the variable names or type one in. This attribute is required, and inserts the NAME attribute.

For more information, see "Understanding Scope" on page 120.

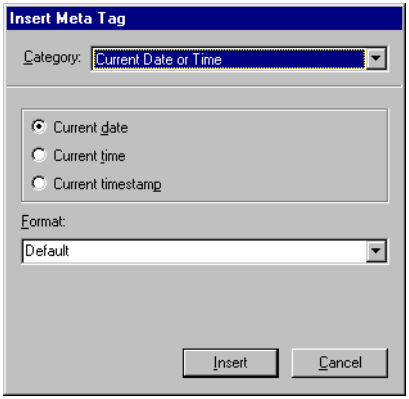
- The **Scope** drop-down list contains:
  - Default
  - Local
  - User
  - Cookie
  - Domain
  - System

  If the scope is other than default, the SCOPE=*selectedScope* attribute is added to the meta tag.

- The **Row** and **Column** fields are enabled only when the **Array element** checkbox is checked. This option adds [*rownumber*, *columnnumber*] to the variable name. If you leave either of the **Row** or **Column** fields empty, the value defaults to "*", which means all rows or columns are returned.

## Current Date/Time

To insert the current date or time using a meta tag, choose **Current Date/Time** from the **Category** drop-down list.




For more information see, "<@CURRENTDATE>, <@CURRENTTIME>, <@CURRENTTIME-STAMP>" on page 65 of the *Meta Tags and Configuration Variables* manual.

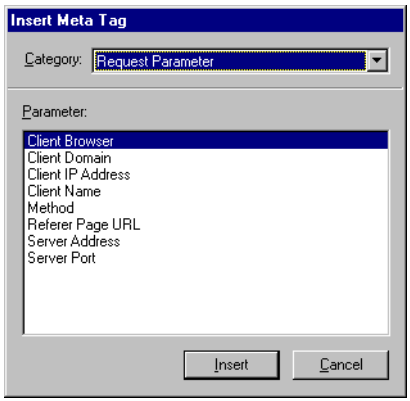
This action inserts `<@CURRENTDATE>`, `<@CURRENTTIME>`, or `<@CURRENTTIMESTAMP>`. There are various options you can set for **Current Date/Time**.

If you select a format other than **Default,** the `FORMAT` attribute is added to the tag.

The **Format** list contains several common formats of the type denoted by the **Current date**, **Current time**, or **Current timestamp** radio button. When the radio button selection changes, the **Format** selection reverts to **Default**.

## Request Parameter

To return a value pertaining to the current user request, choose **Request Parameter** from the **Category** drop-down list.



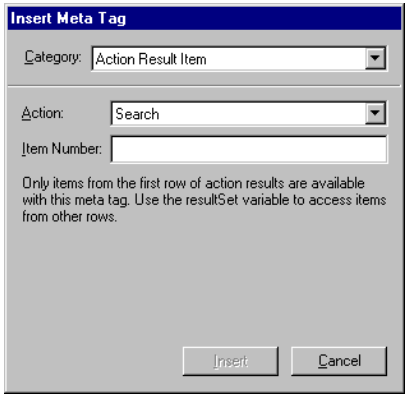

For more information, see "<@CGIPARAM>" on page 50 of the *Meta Tags and Configuration Variables* manual.

The **Parameter** list includes items corresponding to all of the `<@CGIPARAM>` tag parameters.

This action inserts `<@CGIPARAM NAME=`*paramName*`>`, where *paramName* is the CGI parameter name corresponding to the selected item in the list.

## Action Result Item

To insert a meta tag that returns values from the first row of results for previously executed actions in the current application file execution, choose **Action Result Item** from the **Category** drop-down list.



For more information, see "<@ACTIONRESULT>" on page 26 of the *Meta Tags and Configuration Variables* manual.

Action result items specified here are data from the first row of results generated by the action. A Search action, for example, may return 100 rows of data in ten columns. Specifying action result item six from that action (`<@ACTIONRESULT searchActionName 6>`) gives you the value from row one, column six.

CHAPTER SEVEN

# Working With Variables

7

## How to Use Variables

*Variables* are placeholders that you can assign a value to; they are created and assigned values using the Assign action or the `<@ASSIGN>` meta tag.

Every variable belongs to a *scope*, which tells Tango if the variable is to be used only for the particular application file execution, for a user, or for a particular domain being served with Tango.

*Arrays* are a special variable type that allow you to create a structured data table with multiple values, as opposed to standard variables which only store one value.

One important set of variables determines the behavior of certain Tango options. These are called *configuration variables.*

This chapter covers the following topics:

- introduction to variables—standard and array—including variable scope and its effects
- how to assign values to variables
- configuration variables
- changing a special user variable option: the user key.

# About Variables

Variables are defined and given values with an Assign action in a Tango application file.

For more information, see “Where You Can Use Meta Tags” on page 111.

Variables can also be assigned values by using the <@ASSIGN> meta tag.

You can assign any combination of literal values and meta tag values to a variable. For example, to assign a full phone number using values from area code and phone number form fields, you could use "(<@POSTARG area>) <@POSTARG phone>" as the variable value.

## Naming Variables

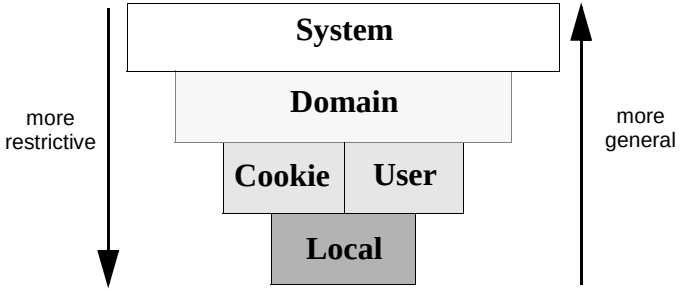
All variable names:

- must start with a letter
- may contain numbers, letters, and the underscore (“_”) character.
- may be no longer than 31 characters.

Variable names are case insensitive; for example, myVar is the same variable as MYVAR and MyVaR.

## Understanding Scope

Every variable belongs to a *scope*. There are several scopes that are used for different purposes within Tango. The following diagram shows the different scopes and their relationships.




Variables that are assigned values in more restrictive scopes override variables with the same name that are assigned values in more general scopes.

*Local* scope is the most restrictive; variables that belong to this scope are used only for the execution of a particular application file.

*Cookie* scope refers to variables that are sent to and kept by Web browsers. *User* scope refers to variables that are set for particular users within Tango. *Domain* scope refers to variables that are used in a particular World Wide Web domain. *System* scope is the most general, applying to the entire Tango Server. This scope is restricted to configuration variables.

You can find more details on these scopes in the following sections.

## Local

*Local* scope is used for variables that expire after an application file is executed. That is, after the application file (and its branches, if any) have finished executing, the variable is purged from memory.

Local variables could be used for a counter within a loop. The counter is not needed outside the application file that it is in, so using a local variable would be appropriate.

For another example, a user might enter a choice during the execution of an application file to have full explanations returned during the execution of an application file (for a beginning user), or rather terse commands (for an experienced user). At the beginning of the application file, a local variable is set, and that variable is used in the rest of the application file to determine what HTML is returned to the user.

## User

Variables that have *user* scope let you store and access values associated with a particular user. Once an assignment has been made to a user variable, its value is available in subsequent actions within the same application file execution and in any application files executed afterward.

For example, user variables would be necessary in an application file that logs users in to a private area of your Web site, displaying a Web page that prompts for a user's name and password. You might want to save the entered name so you can display it on personalized pages in subsequent queries. Or you may need to use the name to restrict database queries performed by that user. Both of these actions would be performed with a user variable.

For more information, see "variableTimeout" on page 191 of the *Meta Tags and Configuration Variables* manual.

User variables expire 30 minutes after the user associated with them last accesses Tango. You can change the timeout by changing the value of the Tango configuration variable `variableTimeout`.

## Cookie

*Cookie* scope is used for variables that are sent to the user's browser as cookies (that is, saved in a small text file and kept by the browser for a specified amount of time).

Cookies are saved by the browser and then are sent to the site by the browser when it returns to the site that generated the cookie in the first place. Cookie variables can be used to save state information between browser sessions or in a single browser session.

For more information, see "Cookie Properties" on page 132.

There are a variety of cookie options that can be set for cookie variables, such as when the cookies expire.

## Domain

*Domain* scope is used for variables that are relevant to a particular domain. Domain variables, like system variables, are persistent across application files and users.

For more information, see "variableTimeout" on page 191 of the *Meta Tags and Configuration Variables* manual.

Domain scope variables, by default, expire after the time period specified by `variableTimeout` (system scope). To change the expiry timeout period for domain variables, assign the desired value to `variableTimeout` in domain scope.

Domains are different base URLs; for example, one Web server could be hosting several different domains (`www.my_company.com`, `www.your_company.com`, `www.a_non_profit.org`). Tango uses an encrypted form of the domain used to access the file as the domain key, and any variable set with `scope=DOMAIN` is set for the particular domain where the application file is executed.

**Caution** You can add your own password to the encrypted `domainScopeKey` if you want to ensure that application files created by others and run on your server cannot change your domain variables.

For more information, see "domainScopeKey" on page 175 of the *Meta Tags and Configuration Variables* manual.

For example, a variable called `organization` with domain scope could be assigned a value in two different domains hosted on the same Web server (values are "My Company, Inc." and "My Non-Profit"). When an application file returns the value of `organization SCOPE=domain` (say, in a standard footer), a different value is returned depending on the URL that is being accessed, even though the variable name is the same.

Domain scope for variables lets you easily store values that are shared by all users of a particular domain. By default, the domain key (that is, the piece of information Tango uses to determine which domain a user belongs to) is based on the SERVER_NAME CGI parameter. The value of the SERVER_NAME parameter comes from the URL used to initiate each request by a user. For example, if a user requests

```
http://www.yourserver.com/foo.taf
```

the domain variable is keyed on "www.yourserver.com". Everyone using "www.yourserver.com" to access a Tango application file gets the same values for domain variables.

However, if a user accesses the exact same application files with the IP address of the server (http://206.186.95.106/foo.taf), the domain scope variable is different because the key value is now the server's IP address not the server's name. The same applies if you access the Web site locally with:

```
http://localhost/foo.taf
```

This behavior is not usually a problem because most users access your site with the server name, not its IP address. If, however, you need to be sure domain variables are being shared by users hitting your site with its IP address and also using its domain name, consider the following options:

- If your Tango Server is hosting just one domain, you can set the domain scope key to the name of that domain (for example, "www.myserver.com" or any constant value). That way, the value of the key is the same for everyone using your server, regardless of the URL used to access it. Use the configuration application file (config.taf) to set the value of the domainScopeKey variable to the desired value.

For more information, see "httpHeader" on page 177 of the *Meta Tags and Configuration Variables* manual.

- At the main entry points of your site, redirect the user to your site using the domain name. You can use a Tango application file to do this by setting the httpHeader variable (local scope) to:

  ```
  HTTP/1.1 302 Temporarily Moved<@CRLF>Location:
  http://www.yourserver.com/foo.taf<@CRLF><@CRLF>
  ```

For more information, see "<@CRLF>" on page 62 of the *Meta Tags and Configuration Variables* manual.

- Use absolute URLs in links to Tango application files. For example, instead of <A HREF="/mydoc.taf">mydoc</A> use <A HREF="http://www.myserver.com/mydoc.taf">mydoc</A>

## System

*System* scope is used for variables that are set at the system level, that is, only configuration variables. Many configuration variables that affect the behavior of Tango are set with system scope. For example, the variable `dateFormat` sets the way the date is displayed when it is returned by Tango with a meta tag such as `<@CURRENTDATE>`. If this variable is set with system scope, then all dates returned by Tango are in that format.

For more details on different scopes for configuration variables, see "Understanding Scope" on page 120.

Certain configuration variables can be set for different scopes. This means that the value changes in one particular scope (and all scopes that scope dominates) while maintaining its default value elsewhere.

For example, `dateFormat` can be set with `scope=LOCAL`, which changes the format of the date for the current application file; `scope=USER`, which would change the format of the date for all application files executed by a particular user; or `scope=DOMAIN`, which would change the format of the date for all application files and users that use a particular URL containing a domain to access Tango.

### *To set system configuration variables*

For more information, see "Configuring Tango Server" on page 317.

You can set system configuration variables with the `config.taf` application file. This application file prompts you for a password and then presents groups of related configuration variables on different screens, allowing you to easily set system configuration variables.

To set configuration variables without using the `config.taf` application file (that is, in an application file), you must know the correct password. This password is stored in the `t3server.ini` configuration file and a configuration variable called `configPasswd`.

When you attempt to set a system configuration variable, Tango checks to see if there is a variable called `configPasswd` with `scope=USER` that matches `configPasswd` with `scope=SYSTEM`. If there is, Tango lets you change configuration variables. If not, Tango returns an error message.

Setting `configPasswd scope=USER` can interact with user keying mechanisms; see "Changing the User Key" on page 142..

That is, you must assign the value of an entered string (for example, `<@POSTARG NAME="Password">`) to the configuration variable `configPasswd` with `scope=USER` using the Assign action or the `<@ASSIGN>` meta tag.

## Returning Variable Values

To have Tango return the value of a particular variable, use the <@VAR> meta tag. This tag is replaced with the variable value at the time that the application file is executed. For example, to return the value of the variable named `fred` in user scope:

```
<@VAR NAME="fred" SCOPE="user">
```

### Default Scoping Rules

If you do not specify a scope parameter, Tango checks for matching variables in different scopes, from the most restrictive to the most general:

```
<@VAR NAME="fred">
```

Tango checks for matching variables in local, user, cookie, domain, and system scope, returning the value for the first matching variable that it finds.

**Note** Cookie scope is a special scope, and Tango does not check for matching variables with this scope when returning unscoped variable values with <@VAR>.

### Shortcut Syntax for Returning Variables (@@foo)

There is a shortcut syntax for returning variables as well, with or without scope: use a double "@" and the name of the variable. The following two notations are equivalent:

```
<@VAR NAME="fred"> ⇔ @@fred
```

To use a scope parameter with this shortcut, place it in front of the variable being accessed, with a dollar sign ("$") in between. The following two notations are equivalent:

```
<@VAR NAME="fred" SCOPE="user"> ⇔ @@user$fred
```

If you are creating complicated meta tag syntax, using this shortcut may help to make things clearer.

The 31-character length limit on variable names is exclusive of any scope specifier; for example, in `local$longvariablename`, the length limit applies to the variable name portion.

## Arrays

Arrays are a special type of variable that allow you to store many different values in a structured format. This is distinct from the standard variable, which only stores one value.

Arrays are structured as a table with rows and columns; values are saved at each row and column intersection. This is similar to the way values are saved in a database table: as rows and columns.

For example, a three-row by four-column array with the values of the first twelve integers would look like this:

```
1  2  3  4
5  6  7  8
9  10 11 12
```

### Setting Arrays

For more information, see "<@ARRAY>" on page 32 of the *Meta Tags and Configuration Variables* manual.

To create an array—for example, the one in the previous paragraph—use the <@ARRAY> tag within the Assign action or the <@ASSIGN> and <@ARRAY> meta tags together:

You can use the <@ARRAY> tag to create arrays by using only rows and columns (ROW attribute and COLS attribute), which creates an array with the specified dimensions, all of whose elements are empty, or by using the VALUE attribute to create and initialize the array with values. If ROWS, COLS, and VALUE are specified, they must match in terms of the number of rows and columns specified.

The meta tag assignment of an array to a variable looks as follows:

For more information, see "<@ASSIGN>" on page 35 of the *Meta Tags and Configuration Variables* manual.

```
<@ASSIGN NAME="arrayVar" VALUE=<@ARRAY ROWS="3"
COLS="4" VALUE="1,2,3,4;5,6,7,8;9,10,11,12">>
```

When using an Assign action to create an array and assign it to a variable, the **Value** field would contain an <@ARRAY> meta tag.

For more information, see "cDelim" on page 167. and "rDelim" on page 184 of the *Meta Tags and Configuration Variables* manual.

(The row and column delimiters for VALUE are set with the configuration variables rDelim and cDelim, or with the <@ARRAY> tag's optional attributes RDELIM and CDELIM. By default, they are set to ";" and "," respectively.)

Array cells can contain single values only. They cannot contain other arrays.

### Array Formats

How arrays are returned depends on context, that is, where an array-returning meta tag such as <@VAR> is used.

Arrays are returned as array variables (with their special internal structure) when used in assignments to other variables, for example:

```
<@ASSIGN NAME="fred" VALUE="<@VAR NAME=
'array_variable'>">
```

When referred to anywhere else (for example, returned in Results HTML using the <@VAR> meta tag), arrays are converted to a text representation using the supplied <@VAR> attributes for prefixes and suffixes, or the configuration variable defaults, if no attributes are specified.

## Returning the Values of Arrays

Remember an array is *not* just a list of values; it has two-dimensional structure. For example, if you set up and gave values to the variable fred as above and then asked Tango to return the value of the array in HTML only, Tango would return the structured string of values, using delimiters to separate rows and columns.

Along with many other uses for programming and database work, arrays offer a quick method of returning a table or ordered list of values in HTML. The <@VAR> meta tag has attributes that can be used with arrays:

APrefix: the array prefix string
ASuffix: the array suffix string
RPrefix: the row prefix string
RSuffix: the row suffix string
CPrefix: the column prefix string
CSuffix: the column suffix string

For detailed information on all configuration variables, see the *Meta Tags and Configuration Variables* manual.

The defaults of these parameters are set with configuration variables. If the above array variable is returned using <@VAR> with the default parameters:

```
APrefix='<TABLE BORDER=1>'
ASuffix='</TABLE>'
RPrefix='<TR>'
RSuffix='</TR>'
CPrefix='<TD>'
CSuffix='</TD>'>
```

the following simple table is returned when <@VAR NAME="fred"> is retrieved in Results HTML:

```
<TABLE BORDER=1><TR><TD>1</TD><TD>2</TD><TD>3</TD>
<TD>4</TD></TR><TR><TD>5</TD><TD>6</TD><TD>7</TD>
<TD>8</TD></TR><TR><TD>9</TD><TD>10</TD>
<TD>11</TD><TD>12</TD></TR></TABLE>
```

For more information on arrays, see "<@ARRAY>" on page 32, "<@VAR>" on page 154, and other meta tags that return arrays in the *Meta Tags and Configuration Variables* manual.

Using different suffixes and prefixes could create different kinds of HTML code, such as various kinds of lists.

You can retrieve one single value from an array by specifying row and column co-ordinates:

```
<@VAR NAME="FRED[2,3]">
⇒ Tango returns 7
```

You can return one column or one row from an array by specifying only one of the co-ordinates, and setting the other to *:

```
<@VAR NAME="FRED[2,*]">
⇒ Tango returns a one-column array with the values 5 6 7 8
<@VAR NAME="FRED[*,3]">
⇒ Tango returns a one-row array with the values 3 7 11
```

## Special Array: resultSet

Whenever an action returns a result rowset in Tango (for example, a Search action), that rowset is assigned, in array format, to the local variable `resultSet`. This variable could be used in Results HTML to return your search result, for example:

```
Your search returned the following results:
<@VAR NAME="resultSet" SCOPE="local">.
```

As with any array variable, you can use the prefix and suffix attribute name/value pairs to change how the array is formatted in HTML.

### *resultSet Named Columns*

A special property of `resultSet` is that when it returns the results of a Search action or Direct DBMS query, its columns are *named*. This means that you can refer to the names of columns (instead of the column number) when returning the value of the `resultSet`

array. For example, if you selected these columns in the order shown (as the result of a Search action):

| customer_last_name | customer_ID | phone_number |
|---|---|---|
| Smith | 8099 | 555-1155 |
| Jones | 3334 | 555-1454 |
| Johnson | 1234 | 555-0023 |

The following expressions in Results HTML evaluate as shown:

```
<@VAR NAME="resultSet[3,2]">
```
⇒ Tango returns `1234`

Using named columns, you could return the same values using the following syntax:

```
<@VAR NAME="resultSet[3,customer_ID]">
```
⇒ Tango returns `1234`

You can use a combination of asterisks and named columns:

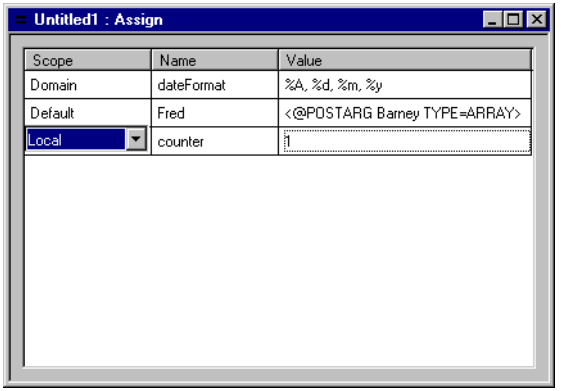
```
<@VAR NAME="resultSet[*,customer_Name]">
```
⇒ Tango returns `Smith Jones Johnson` (a one-column array).

# Assigning Variables With the Assign Action

### *To assign a variable*

Drag the Assign action from the Action bar into the application file window.

The Assign action is inserted into the application file and the Assign editing window appears.




The Assign editing window consists of a three-column list. Each row of the list shows the variable scope, name and value, along with a field that changes to a pull-down menu when active. This field allows you to change the scope. The scope can be one of **Local**, **Cookie**, **User**, **Domain**, or **System**.

You can add one or more variable assignments to each Assign action.

### *To add a new variable assignment to this window*

Do one of the following:

- From the Edit menu, choose **Insert**.
- On the main toolbar, click the Insert icon.
- Right click the Assign window and choose **Insert Assignment** from the context-sensitive menu that appears.

### *To select an assignment*

Click the row.

### *To move an assignment*

Click an assignment and drag it to the desired position.

### *To delete an assignment*

1. Click the assignment you want to delete.
2. Do one of the following:
   - From the **Edit** menu, choose **Delete**.
   - On the main toolbar, click the Delete icon.
   - Press the DELETE key.
3. When the dialog box appears asking you to confirm the deletion, click **OK**.

You can also use the right mouse menu in the Assignment window. See "Editing Variable Assignments" on page 131.

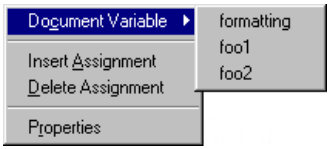
## Editing Variable Assignments

For more information, see "defaultScope" on page 175 of the *Meta Tags and Configuration Variables* manual.

You can type text in the **Name** and **Value** fields, and assign scope in the **Scope** field.

The **Scope** field contains a pop-up menu which allows you to change the scope. The scope can be **Default**, **Local**, **Cookie**, **User**, **Domain** or **System**.

Using the context sensitive menu, you can fill in names from previously assigned variables.




The **Document Variables** submenu contains an alphabetical list of all variables assigned (via Assign actions) in the current application file. Selecting an item from this list sets both the scope and name for the variable assignment.

## Context Sensitive Menu

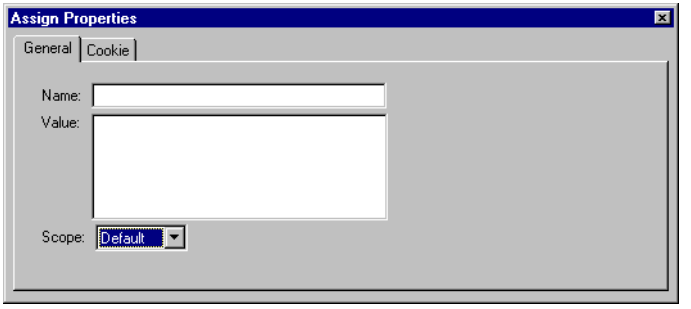
The content area of the Assignment rows within the Assign window supports a context-sensitive menu, as seen above.

The menu contains **Document Variable**, **Insert Assignment**, **Delete Assignment**, and **Properties** items. All of these are active when you invoke the menu on an assignment. If you click a selected row, then the right click menu contains just **Delete Assignment** and **Properties** items.

## Editing Properties

Selecting the **Properties** command when an assignment is selected (either the whole line or while editing any part of it), or selecting an assignment when the properties window is already open shows the assignment's properties.

The Assign Properties window consists of two tabs: General and Cookie. The General tab allows editing of the name, value and scope.

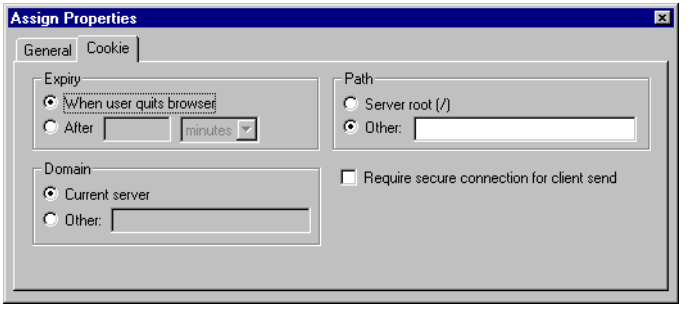



The Cookie tab appears only if the scope for the current assignment is "Cookie".

### *Cookie Properties*

The Cookie section contains controls for editing attributes of cookie variables. (Each group of settings corresponds to an element of the Set-Cookie HTTP header line.)

The default values for new cookies are as shown.




- **Expiry**

  **When user quits browser** is the default cookie behavior as described in the cookie specifications. When this option is chosen, the `Expires` value is omitted from `Set-Cookie` line in the cookie header.

**After __ [time units].** The drop-down list for time units has minutes, hours, days, and years options. The text entry field holds up to 31 characters and a meta tag can be specified there.

- **Domain**

  **Current server** omits the Domain value from the `Set-Cookie` line, causing the cookie to be valid for the current server.

  **Other** allows specification of any domain string up to 31 characters. "`.example.com`", for example, would cause the cookie to be sent back to `www.example.com`, `demo.example.com`, `sales.example.com`, and so on.

- **Path**

  **Server root (/)** specifies that the cookie be sent for all paths within the specified domain.

  **Other** allows specification of a path string up to 31 characters. For example, `/tango/` would cause the cookie to be sent back only for URLs below the `tango` folder.

- **Require secure connection for client send**

  **True** (enabled) or **False** (disabled). This option sets the Secure value of the `Set-Cookie` line. If the value is set to true, then the cookie is sent back by the browser only if a secure connection is being made.

## How Tango Determines Default Scope in Variable Assignments

When you assign a value to a variable but do not specify a scope, Tango performs these steps to determine which scope to use:

- Tango looks for an existing variable with that name. The search starts in the **local** scope, and continues up through **user**, **cookie**, **domain**, and finally to **system** scope. If Tango finds the variable, it assigns the value to it and stops looking.
- If no variable with the specified name is found, Tango creates the variable in the user scope. (This default scope for new variables can be changed using the `defaultScope` configuration variable).

Here are some examples. Assume these variables are already defined:

| Name | Scope |
|---|---|
| foo | local |
| doh | domain |
| ipsum | user |
| lorem | domain |

`<@ASSIGN NAME="foo" VALUE="myVal">` assigns `myVal` to the existing local variable `foo`.

`<@ASSIGN NAME="doh" VALUE="myVal">` creates a new user variable called `doh` and assigns `myVal` to it.

`<@ASSIGN NAME="ipsum" VALUE="myVal" SCOPE="local">` creates a new local variable called `ipsum` and assigns `myVal` to it.

`<@ASSIGN NAME="lorem" VALUE="myVal">` assigns `myVal` to the domain variable `lorem`.

# Using Configuration Variables

Configuration variables are special values that control basic Tango behaviors. For example, there are configuration variables for controlling such settings as:

- the type of information written to Tango Server's log file (`loggingLevel`)
- the default date format used by Tango Server (`dateFormat`)
- how long user variables last before expiring (`variableTimeout`).

For a complete and detailed list of configuration variables, see the *Meta Tags and Configuration Variables* manual.

Some configuration variables can be set in all scopes (except cookie scope), some in particular scopes, and some only in system scope. For those configuration variables that can be set in all scopes, the different scopes have the following effects:

For more information, see "Configuring Tango Server" on page 317.

- **`scope=SYSTEM`** affects *all* application files executed by Tango Server. Assignments made to these configuration variables remain in effect until you change them again (even after stopping and starting Tango Server). The values of these variables are saved in the `t3server.ini` configuration file.

  For more information on how to use `config.taf`, see Chapter 20.

  The `config.taf` application file provided with Tango (in the `admin` folder) makes it easy to change the values of these configuration variables from your Web browser application. You need to know the password set by `configPasswd` in order to set configuration variables for the system.

  For example, the configuration variable `dateFormat scope=SYSTEM` would set the format of the date returned by such meta tags as `<@CURRENTDATE>`.

- **`scope=DOMAIN`** affects the configuration variables in a particular World Wide Web domain.

  For example, there are situations where it is useful to have a different `dateFormat` in a different domain. A company that does business in French and one that does business in English are being hosted on the same Web server. Because of the different conventions for date formats in the different languages, they would want the date to look quite different in each domain.

The configuration variable `dateFormat` with `scope=DOMAIN` would be set to different values within each domain, and from then on, dates returned by Tango (with a meta tag such as `<@CURRENTDATE>`) would have different formats in the different domains.

- **`scope=USER`** affects application files executed by a particular user. As with normal user variables, these depend on the setting of a reliable user key in order to work as expected, and they expire 30 minutes after that user last accesses Tango.

  For example, you could offer the user the chance to set the `dateFormat scope=USER` variable, and, for that particular user from that point on, dates would be formatted the way that user wants.

- **`scope=LOCAL`** affects a particular Tango application file execution, to override the system, domain, or user settings in a particular case. The change is effective from the action in which you make the assignment until the end of the application file's execution (including all its branches if it branches to another application file). You could change the `dateFormat` for a particular application file, for example.

For more information, see "userKey, altuserKey" on page 189 of the *Meta Tags and Configuration Variables* manual.

For more information, see "Changing the User Key" on page 142.

For more information, see "Assigning Variables With the Assign Action" on page 130.

You assign values to configuration variables in the same way as assignments to other kinds of variables: by using Tango Editor's Assign action or by using the `<@ASSIGN>` meta tag to set configuration variables in HTML processed by Tango.

**Caution** Users cannot set system-level configuration variables unless they know the administrative password.

As with Tango's meta tags, letter case is not important when referencing configuration variables. For example, to Tango, `cacheSize`, `cachesize`, and `CACHESIZE` all represent the same configuration variable.

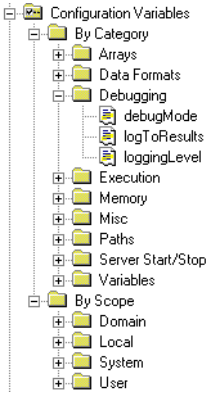
# Shortcuts to Configuration Variable Assignments: Snippets

The Workspace contains configuration variable snippets which create assignments for configuration variables. The configuration variable snippets are organized alphabetically by category (Arrays, Data Format, Debugging, and so on) and by Scope (Domain, Local, System, and User). This provides you with a quick way of creating assignments to configuration variables by dragging these snippets into an Assign action window or an HTML editing field.

For more information, see “Using Snippets” on page 97.

### *To use the configuration variable assignment snippets*

1. Open the Snippets Workspace by showing the Workspace and clicking the **Snippets** tab.
2. Expand the **Configuration Variables** folder.




   A tooltip appears if you place your cursor over the snippet, giving the content of the snippet.

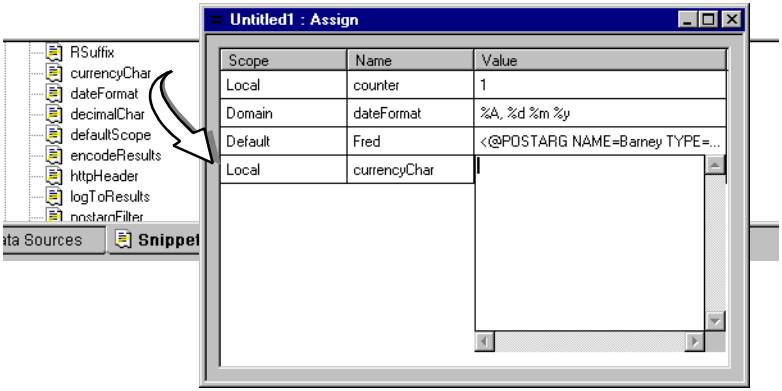
3. Choose the configuration variable to which you want an assignment created and drag in into the HTML or Assign action window.

For more information, see “Using Snippets” on page 97.

Dragging a configuration variable snippet into the HTML window creates an assignment using meta tags, for example, `<@ASSIGN NAME="currencyChar" SCOPE="Local" Value="¥">`. Tango puts your cursor at the special `¥` symbol, so you can just start typing the value of the configuration variable.

When you drag a snippet into an Assign action window, the configuration variable assignment is automatically created for you.

The **Value** field becomes active, ready for you to type a value in:




If you choose a snippet without a defined scope (that is, from the category section of the configuration variable snippet), the scope is set to Default.

# Using User Keys

To associate a user variable with a particular user, Tango must have a piece of information that it can use to uniquely identify that user.

Tango refers to the unique identifier used for tracking a user's variables as the *user key* (formerly known as the *global key*). Tango has several settings allowing you to control what information is used as the user key. Tango default behavior is to use a special meta tag called `<@USERREFERENCE>` as the value of `userKey`.

Under this scheme, when users execute their first Tango application file, Tango generates a unique user reference number and uses it as the user key. In the results sent back to the user, Tango includes the user reference number as an HTTP cookie. This cookie is remembered by the browser and is sent automatically with every subsequent request to your server. Tango checks for the existence of the cookie whenever it accesses a user variable. If it was sent, the cookie value is used as the user key.

To help understand how user variable tracking works, imagine that two users, John and Simone, have each executed an application file that assigns a value to a user variable called `favorite_color`. Tango generates a unique user reference number for each user and sends it in a cookie back to their browsers. The user reference number is a 24-digit hexadecimal string. Inside Tango Server, the user variable information is organized in a manner similar to this:

| User | User Key Value (<@USERREFERENCE>) | Variable Name | Variable Value |
|---|---|---|---|
| John | 4DC3156644A9B130344B6BCC | `favorite_color` | blue |
| Simone | 54A497684AD25940014FDD13 | `favorite_color` | red |

When, in another application file, the user variable `favorite_color` is referenced, Tango first checks to see what the value of the user key is for the current user. It then uses that key value in combination with the user variable name to determine the value to return. If the user is John, the user key value, sent to John's

browser as a cookie, is 4DC3156644A9B130344B6BCC, and the user variable reference returns “blue”; if the user is Simone, the user key value is 54A497684AD25940014FDD13 and it returns “red”.

**Note** User variables may always be used to store information within the span of a single Tango application file execution (that is, from action to action). If you are assigning and referring to a user variable within the same execution, your user variable values will be correct. You may, however, want to use a local variable in that case, which is purged at the end of the execution of the application file. It is only across executions that the user key becomes critical.

## User Keys Specific to Transactions

In the results sent back to the user, Tango includes the user reference number as an HTTP cookie. This cookie is remembered by the browser and is sent automatically with every subsequent request to your server. Cookies are common to the browser application, not specific to a browser window. If a user opens two windows in a browser application, both windows share the same cookies and, therefore, the same user variables. Usually, this is what you want.

Sometimes, however, you want the user variables to be specific to a particular transaction. In this case, you should store the needed values not as user variables but as hidden form fields or search arguments.

Not all browser applications support cookies. Currently, the two major cookie-capable browsers are Netscape Navigator and Microsoft Internet Explorer. If you need to support user reference-based user variables with browsers that do not support cookies, Tango allows the user reference number to be passed via a special search argument, _userReference. The search argument *must* be passed with every URL.

For example:

```
<A HREF="<@CGI>/purchase_item.qry?item_num=
<@COLUMN item_num>&_userReference=
<@USERREFERENCE>">Process Order</A>
```

There is also a shortcut meta tag for including the entire search argument, <@USERREFERENCEARGUMENT>.

Using this tag, the URL above can be rewritten as:

```
<A HREF="<@CGI>/purchase_item?item_num=
<@COLUMN item_num>&<@USERREFERENCEARGUMENT>">
Process Order</A>
```

When using user variables keyed on the user reference number (the default), it is important that all application files have this search argument passed in. If it is missing, and the browser does not support cookies, Tango generates a new user reference and uses it as the user key. The old user reference number cannot be retrieved. This means that previously assigned user variables are “orphaned” and references to them evaluated as empty; new user variables are keyed on the newly generated user reference value.

# Changing the User Key

Tango gives you full control over what information is used as the user variable key. It does this via two configuration variables: `userKey` and `altUserKey`.

The contents of `userKey` determines the default key used for tracking user variables. The contents of `altUserKey` with `SCOPE=system` or `domain` determines what key is used when the value of the key specified by `userKey` with `SCOPE=system` or `domain` evaluates to empty. The maximum length of `userKey` and `altUserKey` is 32 characters. As stated above, the default value for `userKey` is `<@USERREFERENCE>`. The default value for `altUserKey` is empty.

**Note** The `userKey` and `altUserKey` specified in the system scope are the default keys. If the domain scope `userKey` and `altUserKey` have been set, their values override the system settings and determine the user key for users in that domain, because of the way that variables are evaluated in Tango.

## Assigning Values to userKey and altUserKey

When you assign a value to `userKey` and `altUserKey`, you must tell Tango Server not to evaluate meta tags in the `VALUE` attribute, but instead to evaluate the meta tag when user variables need to be keyed. This is done with the `<@LITERAL>` meta tag.

The syntax of the assignment to `userKey` of its default value would be as follows:

```
<@ASSIGN NAME="userKey" VALUE="<@LITERAL
VALUE='<@USERREFERENCE>'">
```

For more information, see “<@LITERAL>” on page 101 of the *Meta Tags and Configuration Variables* manual.

For more information, see “<@ASSIGN>” on page 35 of the *Meta Tags and Configuration Variables* manual.

## Alternate User Keys

Here are some alternate possibilities for `userKey` (and `altUserKey`). You must use the `<@LITERAL>` tag when assigning to these configuration variables:

For more information, see "<@CGIPARAM>" on page 50 of the *Meta Tags and Configuration Variables* manual.

- `<@CGIPARAM username>`

  If you are using HTTP authentication for your site or for a particular set of Tango application files, and have each user logging in with a unique user name, this user name can be used to identify users and their user variables.

- `<@CGIPARAM client_ip>`

  Tango can use the client's IP address as a user key. This user keying mechanism is useful when you know that the users hitting your site are guaranteed to have a one-to-one user / IP address mapping

  Unfortunately, in many situations, the IP address is not an accurate method of identifying a particular user. For instance, some corporate networks are set up so that all HTTP requests are routed through a single server. In this case, requests from different users may all have the same IP address. When user variables are keyed on IP address, and an address may represent several users, user variables do not serve their purpose of providing a way to keep user-specific data.

## Returning the Value of userKey and altUserKey

When the `userKey` and `altUserKey` configuration variables are used in Results HTML, they evaluate to the text of the tags, not the tag values. To see the current value of the user key, use the `ENCODING=METAHTML` parameter to the `<@VAR>` tag. For example, if the following text is typed in a Results HTML field:

For more information, see "Encoding Attribute" on page 9 of the *Meta Tags and Configuration Variables* manual.

```
Variables are now being keyed on:
<@VAR NAME="userKey" scope=SYSTEM>.
⇒ Tango returns <@USERREFERENCE>

The value of the key in the current execution is:
<@VAR NAME="userKey" scope="SYSTEM"
 ENCODING="METAHTML">
⇒ Tango returns 4DC3156644A9B130344B6BCC
```

## Using Application File User Keys

You can override the default user key on an application file basis by setting `userKey` and `altUserKey` with `scope=LOCAL`. These work just like their system-wide counterparts, but apply only until the end of the application file execution. Use local user keys when you want to temporarily use a user key different than the system user key.

CHAPTER EIGHT

# Building Actions Using the Tango Builders

8

## How the Tango Builders Work in Application Files

The Tango builders help you create and generate a sequence of actions to perform specific tasks. The builders are added to an application file in exactly the same way you add any other action.

Once you add a builder and configure it, the actions the builder generates are inserted automatically into the application file. Builder information is saved in the application file format, making it cross-platform capable. This means you can use any application file containing a builder across all platforms.

Tango Editor provides two builders.

- The *Search Builder* builds the actions required to perform a search of database records and to update and delete them.

  For details about how to configure the Search Builder, see "About the Search Builder" on page 152.

- The *New Record Builder* builds the actions required to add a new record to a database.

  For details about how to configure the New Record Builder, see "About the New Record Builder" on page 190.

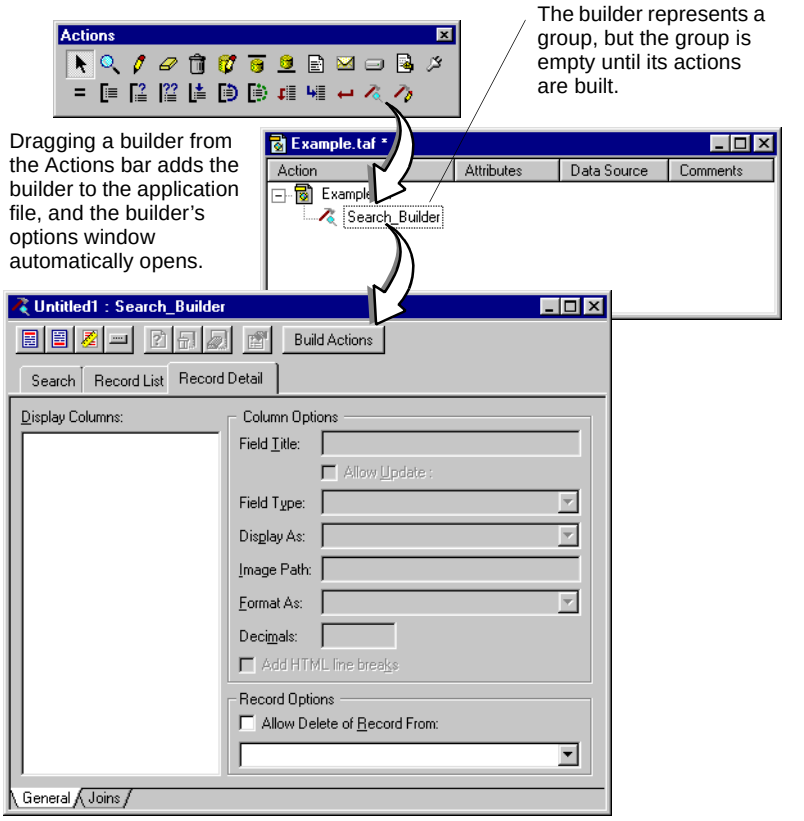
The topics covered in this chapter include:

- adding a builder to an application file
- generating builder actions
- details about the actions generated by the builders.

# Adding a Builder to an Application File

With a new or existing application file open, you add a builder in the same way you add an action, that is, you drag the builder icon from the Actions bar into the application file window.

When you drag either of the builder icons into an application file window, the corresponding builder window opens. For example, if you drag the Search Builder icon from the Actions bar into an open applicile, the Search Builder window opens so you can immediately set Search Builder options.




In the application file window, the corresponding builder icon appears with a default name, just like with actions. The default name for the Search Builder is "Search_Builder" and for the New Record Builder, "Record_Builder".

**Note** Tango Editor can properly process one Search Builder and one New Record Builder in the same application file. If you want to use multiple builders of the same type in a single file, you must handle them using If actions and additional request arguments.

For more information, see “Conditional Action Execution (If Action)” on page 252.

If you add a builder to an application file that already contains the same builder using the default name, Tango adds a number to the default name and increments it by one for each subsequent addition, for example, “Search_Builder1”, “Search_Builder2”, and so on.

**Note** The title of the builder window is in the same form as an action window, `<File name> : <Builder name>`.

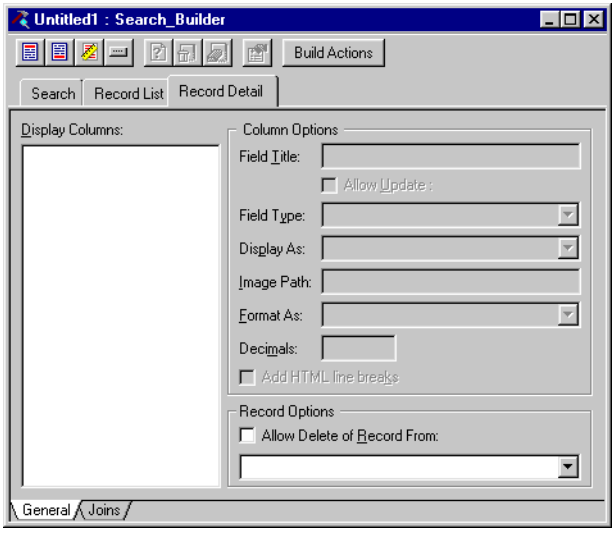
Within the application file, a builder behaves exactly like an action group. That is, it contains actions that can appear in expanded or collapsed form. In the picture on the previous page, the Search Builder icon represents a group, but the group is empty until its actions are built. Tango Server ignores unbuilt builders.

# Building the Actions

Once you have entered all the relevant information in the builder window, you can generate the actions.

### *To build Search or New Record actions in an application file*

**1** Open a builder window (if it is not already open), for example:



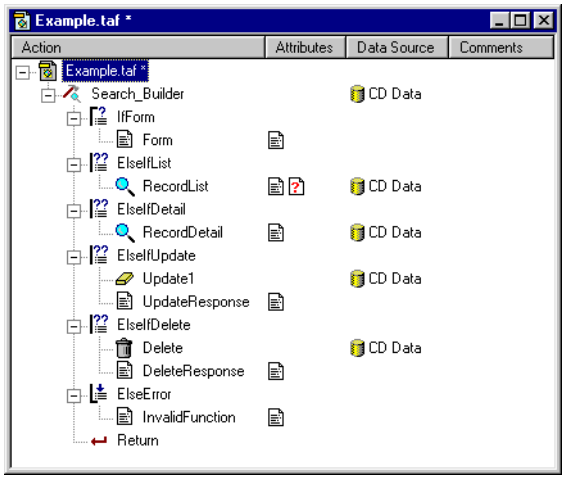


**2** Click **Build Actions**.

The builder checks to see if you have entered all the information required to build the actions.

- If you have not, an error message appears and the build process stops.
- If you have, it builds the actions required to perform the task.

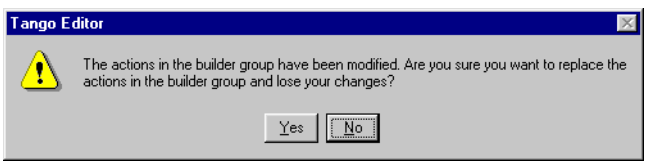
On completion, the actions appear in the builder group. The following diagram shows an application file window after successfully building actions using the Search Builder.




You can view the hierarchy of a builder in an application file. Click either the plus sign (+) or minus sign (-) to expand or collapse the builder group accordingly.

When building actions, the builder replaces any existing actions in the group:

- If the builder group is empty, the actions are inserted into the builder group.
- If the builder group only contains the actions that were generated by the builder and those actions have not been modified, the new actions replace the old actions within the builder group.
- If there are actions in the builder group that the builder did not generate, or the actions in the builder group have been modified, a message box appears.




  - To replace all actions in the builder group with the new actions, click **Yes**.
  - To stop the build action, click **No**.

CHAPTER NINE

# Configuring the Search Builder

9

*Tango Search Builder Options and Setup*

The *Search Builder* builds a series of actions that allows you to create Web forms that can be used to search a database, display the results of the search, and view, update, and delete individual records.

The topics covered in this chapter include:

- setting search, record list, and record detail options
- formatting the search form, and record list and record detail pages
- customizing your Web forms and pages, and creating response messages
- tips on modifying how the Search Builder builds actions
- defining joins.

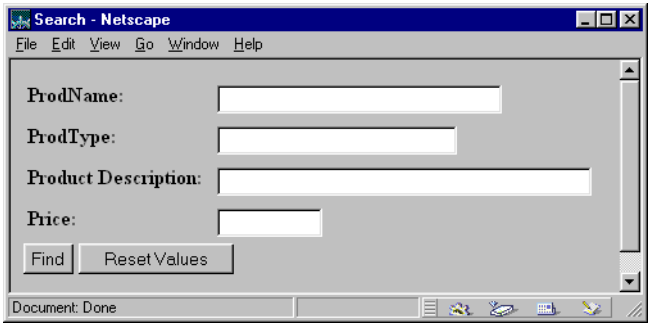
# About the Search Builder

Using the Search Builder you can quickly and easily build the actions necessary to:

- display a form allowing users to specify search criteria
- display a list of matching records
- allow viewing of detailed information on a single record
- allow editing and deleting of the records found.

Based on the settings in each of the option groups and the actions generated, the following are examples of what users might see in their Web browsers.

- On the search form, the user enters the criteria for the records to return.




By default, a **Begins with** search is performed on text columns and an **=** (equals) search on all others. If desired, you can instead let users select each search operator from a drop-down list.

To initiate the search, the user clicks **Find**.

- Tango Server searches the data source for records matching the user's criteria and displays them on the record list page.

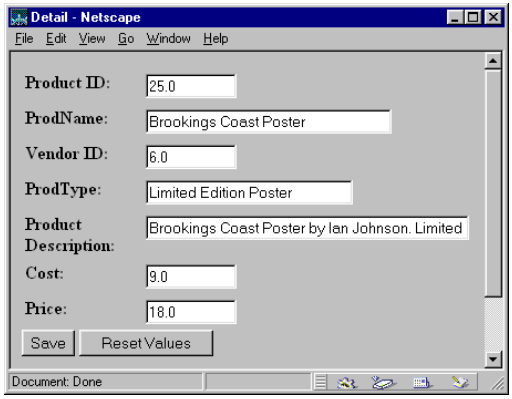

For more information, see "Configuring Tango Server" on page 317.

**Note** Character data from data sources is by default stripped of trailing spaces. You can disable this feature by using the configuration variable `stripChars`: assign the variable the value `false` in local scope for a particular application file, or, if you want to set the variable for all data sources, use the `config.taf` application file to change the variable's value to `false` in system scope.

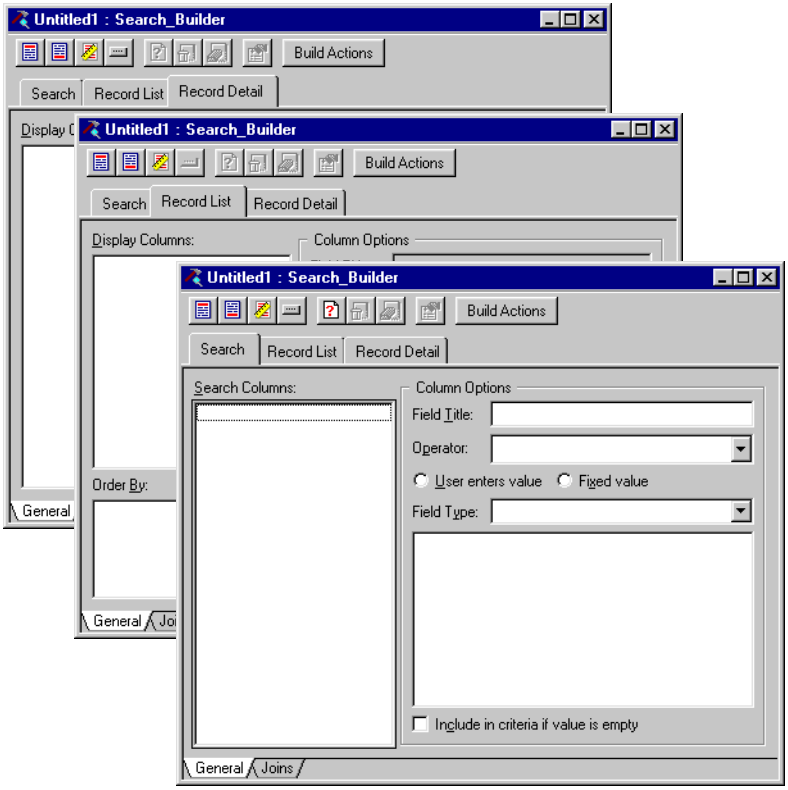
If desired, you can configure the Search Builder to display **Next** and **Previous** buttons for paging through large result sets.

To view detailed information for a record, the user clicks the name of the record in the list, which is hot text, linked to the record detail.

- The record detail page displays more information on the selected record and—if you allow it—lets the user edit or delete it.



When you open the Search Builder window it contains three main groups of options you can specify: Search, Record List, and Record Detail.



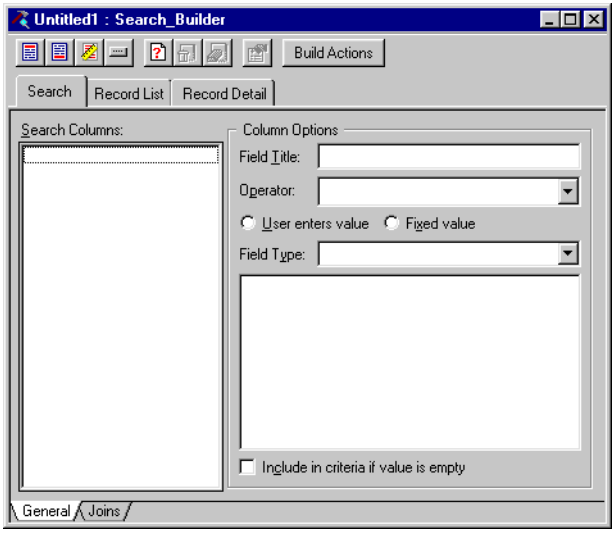

- **Search**. Specify the columns you want users to search and define the format of the search form you want users to see in their Web browsers.
- **Record List**. Specify the columns you want to display in the record list returned to the user after a search and define the format of the Record List Web page.
- **Record Detail**. Specify the columns you want to include on the single-record display page, which appears after a user clicks a specific record in the record list, and define the format of the Record Detail Web page. You can also set up the Record Detail page to allow users to delete the record and/or edit specified columns.

To switch among option groups, click the appropriate tab in the Search Builder window.

# Setting Search Options

Search Builder

When you drag the Search Builder icon from the Actions bar into an application file, the Search Builder window opens, displaying the Search options window.




You use the search options to define the appearance of the search form, which columns the user can search on, and how the values entered by the user are used to search the database. You can also define fixed criteria in addition to the ones the user enters.

## Search Columns List

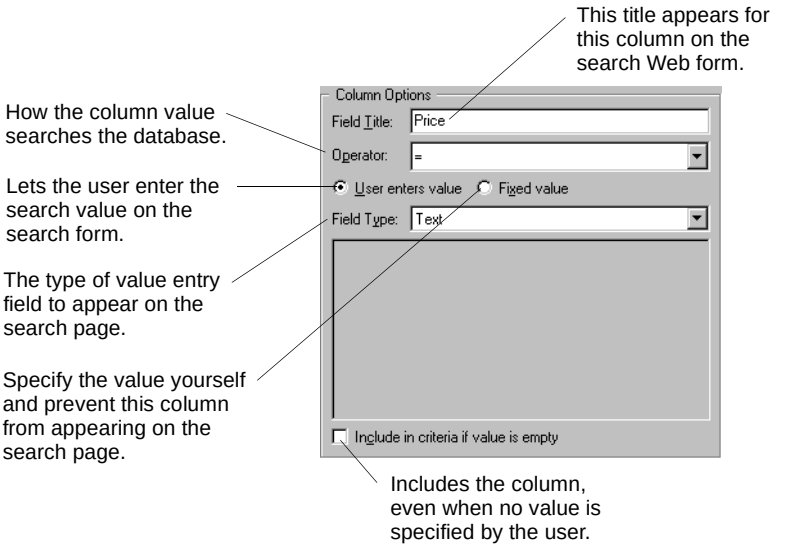
Drag columns from the Data Sources Workspace to this list to use them in defining the search. Columns in the **Search Columns** list appear in the format *table_name.column_name*. The order of the columns in the **Search Columns** list determines the order of the fields on the resulting search form.

The following table describes the operations you can perform on columns.

| To ... | Do This ... |
|---|---|
| Reorder columns | Select the columns and drag them to a different location in the list. |
| Delete columns | Select the columns. Choose **Delete** from the **Edit** menu, press the DELETE key on the keyboard, or select the **Delete Selected Items** icon on the toolbar. |
| Delete columns without confirmation | Hold down the CTRL key while using the **Delete** command. |

## Column Options

Use the **Column Options** section of the Search window to configure each search column. You can specify how each column's entry field appears on the search form and how the value entered by the user is used to search the database.




### Field Title

In **Field Title**, set the title of the value entry field for the column as you want it to appear on the search form.

## Operator

Use this option to set the search operator for the column. For example, if the operator is set to **Begins with**, Tango searches the database for records that begin with the column's search value. If you select **User Enters**, the search form displays a drop-down list of the operators available, and the user can select which operator to use.

## User Enters Value

Select this option if you want the user to enter a value in an entry field on the search form. The available field types are: **Text**, **Drop-down List**, **List Box**, **Check Box**, and **Radio Buttons**.

## Field Type

To select a value entry field type for a column, select the column in the Search Columns list and select an item from the Field Type drop-down list.

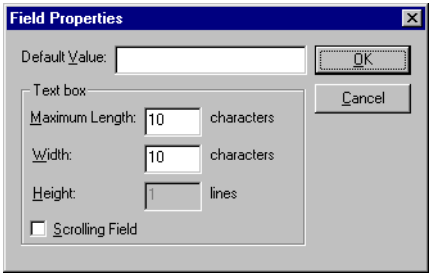
A Field Properties dialog box appears allowing you to specify values for the field.

Field Properties

You can edit a field's values at any time by clicking the **Field Properties** button in the Search Builder window or by choosing **Field Properties** from the **Attributes** menu.

The following describes each of the options in the Field Properties dialog box for each field type:

- **Text**. Use the Text field type to provide single- or multi-line entry of values.




  **Default Value** is the default value you want to appear on the new record entry form.

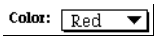
**Maximum Length** is the maximum number of characters the user can enter in the field. This option is not available when **Scrolling Field** is selected because HTML does not support it.

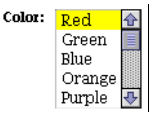
**Width** is the width of the field in characters.

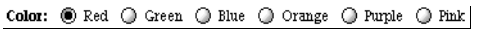
**Height** is the number of lines of text displayed in the field without scrolling. This field is available only when you select **Scrolling Field**. The height of a non-scrolling field is always "1".

**Scrolling Field**. Select this option if you want a multi-line text entry field with a scroll bar.

- **Drop-down List**, **List Box**, and **Radio Buttons.** Each of these field types lets the user select the search value from a predefined list. For example:

Drop-down List 


List Box 

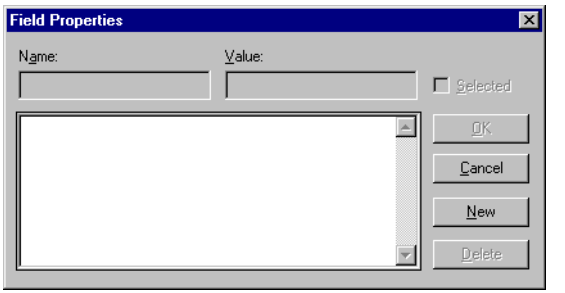

Radio Buttons 


Use the Field Properties dialog box for these field types to specify the values and choose the default. The same dialog box appears for all three field types.

### *To add an item to the values list*

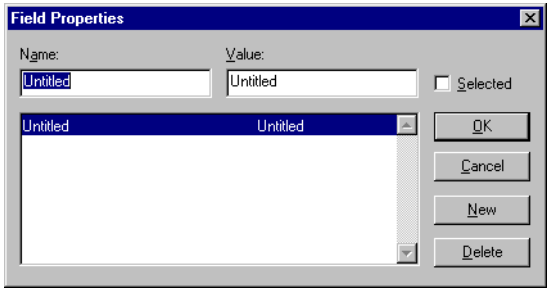
1. Do either of the following:
   - In the Search window, click the **Field Properties** icon.
   - In the main window, choose **Field Properties** from the **Attributes** menu.

   The Field Properties dialog box for the selected column appears.



2 Choose **New**.

An “Untitled” entry appears.




3 Type a name in the **Name** field.

4 Press the TAB key to copy the name into the **Value** field, or enter a value for the item if you want it to be different from the name.

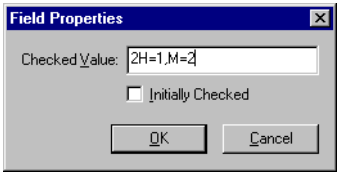
5 Click **OK**.

The **Name** determines what the user sees for this item. The **Value** determines what is used as the search value. If your database column uses abbreviations or codes for values, you can enter a more user-friendly value into the **Name** field and the actual value in the **Value** field. For example, if you are creating a field to search a “state” database column containing abbreviations such as “CA”, “NY”, and “GA”, you can use these values in the list items’ **Value** fields, and the full state names (“California”, “New York”, and “Georgia”) in the **Name** fields.

The following table lists the other operations you can perform on the values list.

| To ... | Do This ... |
|---|---|
| Set which item will be initially selected | Select an item in the list and choose the **Selected** option. The item then appears in bold in the list. If you do not specify an item as **Selected**, the user's browser determines which item is initially selected. Most browsers choose the first item, but some will choose none in this case. |
| Delete an item | Select it in the list, and click **Delete**. To delete without displaying a confirmation dialog box, hold down the CTRL key while deleting. |
| Create an item that causes the column to be omitted from the search criteria when chosen | Leave the **Value** field for that item empty and make sure the **Include criteria if value is empty** option is not selected. |

- **Check Box**. The **Check Box** field type lets the user select between an empty search value (unchecked) and a value you specify (checked).




**Checked Value** is the value to be used for the search if the check box is selected by the user. If the check box is not selected, an empty value is used.

**Initially Checked** specifies whether the check box should be checked by default.

## Fixed Value

When this option is selected, the search value is hard-coded and no entry field appears on the search form. The value you specify is used for every search.

For more information, see "Field Type" on page 157.

Using the **Value** drop-down list, select one of the following options for a fixed value:

- **Value Entered**. Use the text box provided to enter the search value for the column.

- **SQL Expression**. The value returned by the SQL expression text entered is used as the search value. The text entered is evaluated by the database and the result is used as the search value.

**Note** For ODBC data sources, you can enter ODBC scalar functions here.

- **SQL Statement**. The SQL statement entered is executed, the results retrieved, and the first data item of the results is used as the search value. For example, if you enter:

```
SELECT MAX (cust_num) FROM customer
```

  the largest customer number is used as the search value.

- **Current Timestamp**. The current timestamp (date and time combined) on the Tango Server computer is used as the search value.
- **Current Date**. The current date on the Tango Server computer is used as the search value.
- **Current Time**. The current time on the Tango Server computer is used as the search value.
- **CGI Parameters**. The rest of the fixed value options are referred to collectively as CGI parameters. They include **Client Name**, **Client Password**, **Client IP Address**, **Client Browser**, **Server Address**, **Server Port**, **Referer Page URL**, and **Method**. They are passed by either the user's Web browser or the Web server with each request to Tango. When you specify one of these parameters as the search value, the parameter value used is the one passed in when the user clicks **Find**.

For more information on what the CGI parameters evaluate to, see "<@CGIPARAM>" on page 50 of the *Meta Tags and Configuration Variables* manual.

## Summary: Setting Column Options

The following table describes the settings you can make in the Column Options section of the Search window.

| To ... | Do This ... |
|---|---|
| Let the user specify the search value for a column | Select the column in the **Search Columns** list. Select the **User enters value** radio button. This option defines the column as a user-searchable field, and a value entry field will appear on the search form. |

| To ... | Do This ... |
|---|---|
| Specify the title to appear for a column's search form value entry field | Select the column in the **Search Columns** list. The column's name appears in the **Field Title** field. Replace the name with the desired field title. Tango remembers the entered title and uses it as the default the next time you use the column. |
| Specify the type of value entry field to be displayed for a column | Select the column in the **Search Columns** list. Make sure the **User enters value** option is selected. From the **Field Type** drop-down list, select the type of field you want displayed. A Value dialog box appears, allowing you to specify the field attributes. |
| Include a column in the search even if the user leaves its value entry field empty | Select the column in the **Search Columns** list. Select the **Include criteria if value is empty** option. If the user leaves this column's value empty and clicks **Find**, only records that have an empty value in the column are returned. If the option is not selected (the default), the column is omitted from the search when the user does not enter a value. |
| Specify the operator Tango uses when comparing the database column values with the search value | Select the column in the **Search Columns** list. From the **Operator** drop-down list, select the operator that specifies how you would like the column searched. For example, **Begins with** searches for values that begin with the entered value. |
| Let the user select the search operator for a column | Select the column in the **Search Columns** list. Select **User enters** from the **Operator** drop-down list. A drop-down list of available operators appears beside the column's value entry field on the search form. |
| Hard-code the search value for a column | Select the column in the **Search Columns** list. Select the **Fixed value** option. From the **Value** drop-down list, select a search value. You can use one of the preset values, such as current date or time, select **Value Entered** to enter a value yourself, select one of the SQL options to get a search value from the data source, or select a CGI parameter. Columns specified as **Fixed value** do not appear on the search form. |

# Formatting the Search Page

Use the format options to define how the search fields and their titles display.

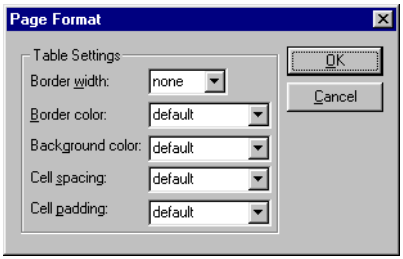
### *To change the format of the search page*

Page Format

Do either of the following to change these options:

- In the Search window, click the **Page Format** icon.
- In the main window, choose **Page Format** from the **Attributes** menu.

The Page Format dialog box appears.




Tango aligns the page fields using an HTML table. Specify the table attributes as follows:

- **Border width**. The width of the table border in pixels. Select **none**, or from numbers **1** to **8**.
- **Border color**. The color of the table border. Select **default** or a color from the list.

**Note** For **Border color**, **Background color**, **Cell spacing**, and **Cell padding**, selecting **default** instead of a value omits that attribute from the HTML and causes the browser's default setting to be used instead.

- **Background color**. The background of the table. Select **default** or a color from the list.
- **Cell spacing**. The amount of space, in pixels, inserted between individual cells in the table. Select **none**, or from numbers **1** to **8**.
- **Cell padding**. The amount of space, in pixels, between the border of a cell and the contents of the cell. Select **none**, or from numbers **1** to **8**.

# Customizing Your Search Form and Creating Result Messages

## Header, Footer, and No Results HTML

Use Header HTML and Footer HTML to customize the search form by specifying HTML to appear above and below the search form.

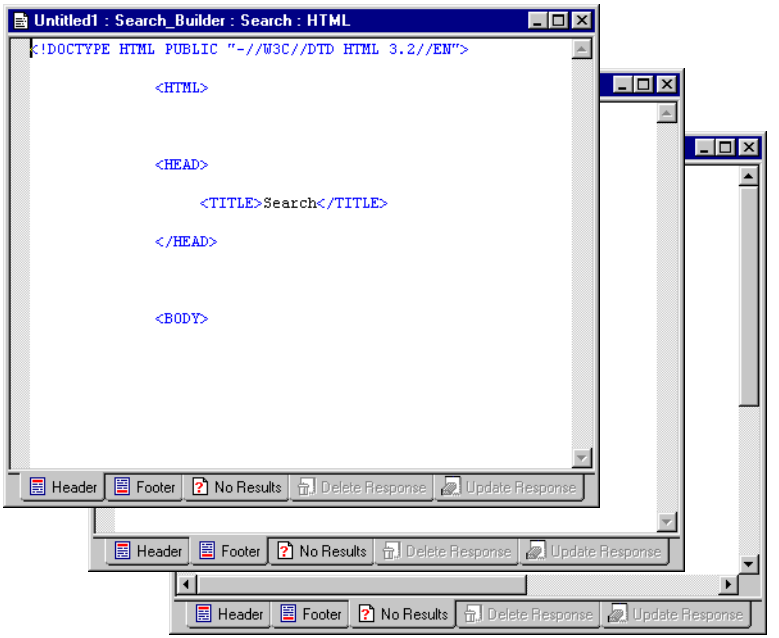
No Results HTML lets you specify the HTML to return when no records match the search criteria specified by the user.


### *To enter Header HTML, Footer HTML, or No Results HTML*

1 Do either of the following:

- In the Search window, click the **Header HTML**, **Footer HTML**, or **No Results HTML** icon.
- In the main window, from the **Attributes** menu, choose **Header HTML**, or **Footer HTML**, or choose **Responses** then **No Results HTML** from the submenu.

The corresponding HTML editing window appears.




You can switch between the HTML editing windows by clicking on the **Header**, **Footer**, and **No Results** tabs at the bottom of the HTML window.

2 Enter the desired HTML.

3 Close the editing window.

## Changing Button Titles

The search form contains two buttons below your search fields:

- The **Reset Values** button resets the entry fields to their default values.
- The **Find** button initiates the search.

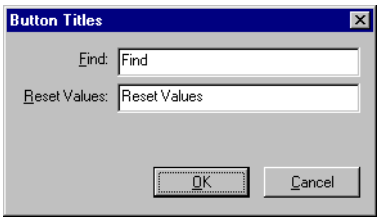
### *To change button titles*

1 Do either of the following:

Button Titles

- Click the **Button Titles** icon.
- From the **Attributes** menu, choose **Button Titles**.

The Button Titles dialog box appears.

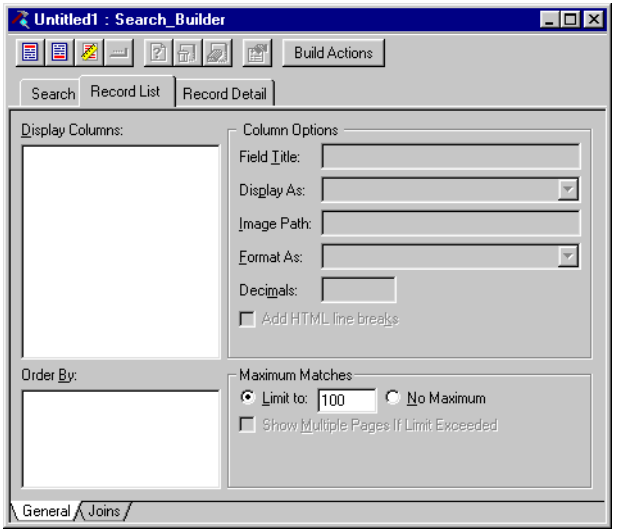



2 Enter new titles in the corresponding fields.

3 Click **OK.**

# Setting Record List Options

Use the options in the Record List window of the Search Builder to define the appearance and functionality of the Web page returned after the search (defined in the Search section) is performed.




Among other things, you can specify:

- which columns from each matching record are displayed
- the ordering of result records
- the maximum number of records to be returned
- whether you want **Next** and **Previous** buttons to appear, allowing paging through large result sets
- which column or columns appear as links to the detail page.

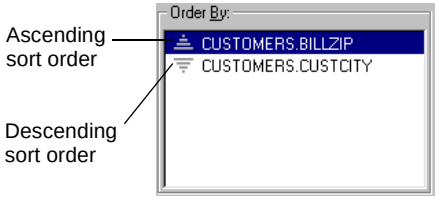
## Display Columns

Drag columns from the Data Sources Workspace to this list to have them retrieved from the database and displayed on the record list Web page. The order in which columns appear in the **Display Columns** list determines their order on the Web page.

## Order By

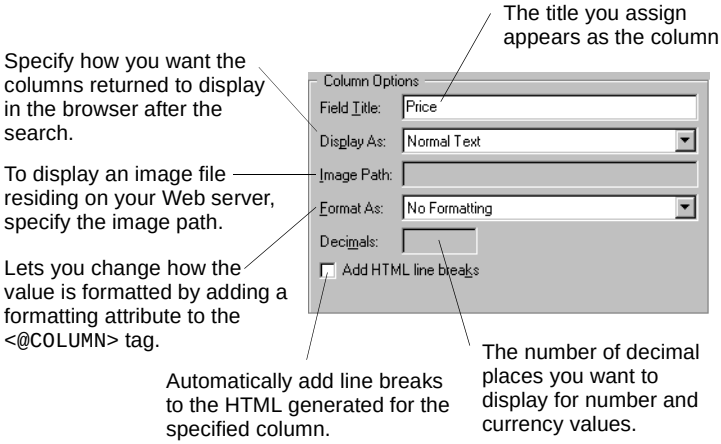
Records from the database are sorted on the record list page according to the order specified in the **Order By** list. You can drag any number of columns into this list; however, each of the columns must also appear in the **Display Columns** list.

The records are sorted by the first column listed. Then records having the same values in that column are ordered by the second column, and so on. The default sort order is ascending, meaning records with lower values in the sort column appear first in the list. You can toggle between ascending and descending by clicking the and icons.




## Column Options

Use the **Column Options** section to set up options for each column in the **Display Columns** list.




### Field Title

In the **Field Title** field, enter the text you want to appear as the column title.

### Display As

You can specify how you want the columns returned to display in the browser after the search. From the drop-down list, select from the following options:

- **Normal Text** adds the following to the HTML generated for the specified column:

  [*columntag*]

  If the **Add HTML line breaks** option is selected for the specified column, the HTML becomes:

  [*columntag* `ENCODING=MULTILINE`]

For more information, see "Encoding Attribute" on page 9 of the *Meta Tags and Configuration Variables* manual.

- **Link to Detail**. Select this option to cause the selected column to appear as a hyperlink to the record detail page, that is, the user can click a value from this column and the detail for that record is displayed. You can specify more than one column as a link to the record detail.

  If you specify no column as a link to the detail page, the first column is automatically chosen for you when actions are built.

- **Link to URL Stored in Column**. If you have a URL stored in your database column, select this option to automatically generate a hot link. This option adds the following to the HTML generated for the specified column:

  `<A HREF="[`*columnvalue*`]">[`*columnvalue*`]</A>`

- **Link to Email Address Stored in Column.** If you have an email address specified in your database column, select this option to automatically generate a `mailto` link. This option adds the following to the HTML generated for the specified column:

  `<A HREF="mailto:[`*columnvalue*`]">[`*columnvalue*`]</A>`

- **Image: File Name Stored in Column**. Select this option to display an image file residing on your Web server. When you select this option, the **Image path** field is enabled in which you enter the path to the image.

  This option adds the following to the HTML generated for the specified column:

  `<IMG SRC="[`*imagepath*`][`*columnvalue* `ENCODING=URL]">`

- **Image: URL Stored in Column.** Select this option to display an image file residing on the Internet; that is, your database stores a URL pointing to the image. This option adds the following to the HTML generated for the specified column:

  `<IMG SRC="[`*columnvalue*`]">`

For more information, see "Encoding Attribute" on page 9 of the *Meta Tags and Configuration Variables* manual.

- **HTML.** Use this option if your database column contains HTML that you would like to display. This option adds the following to the HTML generated for the specified column:

  [*columnvalue* `ENCODING=NONE`]

  If the **Add HTML line breaks** option is selected for the specified column, the HTML becomes:

  [*columnvalue* `ENCODING=MULTILINE`]

## Format As

The **Format As** field is enabled only when you select either the **Normal Text** or **Link to Detail** option from the **Display As** drop-down list.

For more information, see "Format Attribute" on page 11 of the *Meta Tags and Configuration Variables* manual.

Each of the following options in the drop-down list (except **No Formatting**) adds a `FORMAT="formatstring"` attribute to the `<@COLUMN>` tag in the HTML generated for the column in the Record List action's Results HTML.

The following table lists the options and the corresponding format string.

| Option | Format String |
|---|---|
| No Formatting | None |
| Date | `datetime:@@dateFormat` |
| Time | `datetime:@@timeformat` |
| Timestamp | `datetime:@@timeStampFormat` |
| Number with Commas | `num:3-*,'@@thousandsChar',`*decimals*`, '@@decimalChar',,,'-',` |
| Number with No Commas | `,,`*decimals*`,'@@decimalChar',,,'-',` |
| Currency with Commas | `num:3-*,'@@thousandsChar',`*decimals*`, '@@decimalChar','@@currencyChar',, '@@currencyChar(',)` |
| Currency with No Commas | `,,`*decimals*`,'@@decimalChar', '@@currencyChar',,'@@currencyChar(',)` |

### Decimals

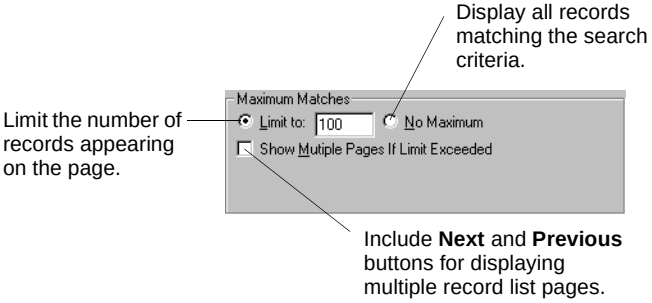
Specify the number of decimal places you want to display for number and currency values. The **Decimals** field is available only when you select one of the number or currency options from the **Format As** drop-down list. The default is **0** for number options and **2** for currency options. An empty or non-numeric value is evaluated as **0**.

### Add HTML line breaks

This option is available only when you select **Normal Text** or **HTML** from the **Display As** drop-down list and **No Formatting** is selected from the **Format As** drop-down list. Otherwise, this option is disabled.

## Maximum Matches

Use the options in this section to restrict the number of matches displayed on the record list page.




### Limit To

Select this option to limit the number of records returned by the search to the number specified. For example, to show only the first 10 records matching the search criteria, select this option and enter "10" in the **Limit To** field.

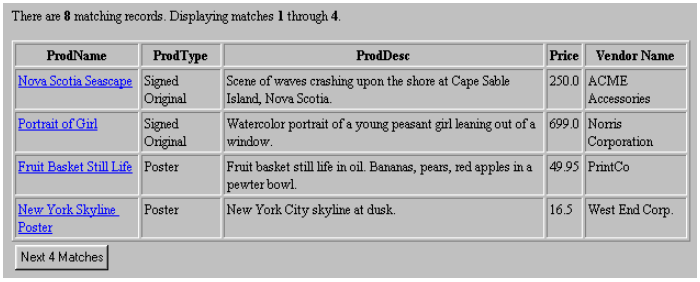
### No Maximum

If you select the **No Maximum** option, all records matching the search criteria are retrieved and displayed on the record list page.

## Show Multiple Pages If Limit Exceeded

If you specify a maximum number of matches in the **Limit To** field, this option is available. If selected, a **Next** button appears on the record list page (if the number of matching records exceeds the limit entered), along with an indication of the total number of records matching and which records are being displayed. When the user clicks the **Next** button, the next group of matching records appears. A **Previous** button appears on record list pages beyond the first, which allows the user to go backwards in the list of matching records.

Selecting the **Show Multiple Pages If Limit Exceeded** option causes result set information and a **Next** button to appear on the results list page.

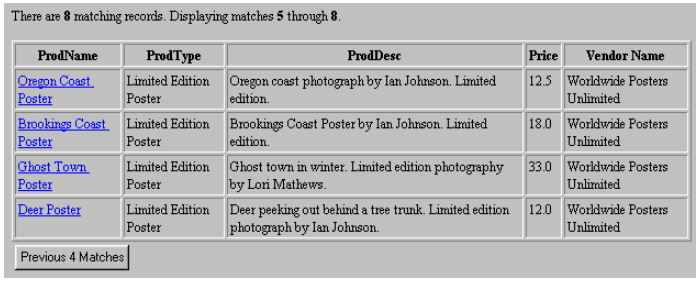


There are **8** matching records. Displaying matches **1** through **4**.

| ProdName | ProdType | ProdDesc | Price | Vendor Name |
|---|---|---|---|---|
| Nova Scotia Seascape | Signed Original | Scene of waves crashing upon the shore at Cape Sable Island, Nova Scotia. | 250.0 | ACME Accessories |
| Portrait of Girl | Signed Original | Watercolor portrait of a young peasant girl leaning out of a window. | 699.0 | Norris Corporation |
| Fruit Basket Still Life | Poster | Fruit basket still life in oil. Bananas, pears, red apples in a pewter bowl. | 49.95 | PrintCo |
| New York Skyline Poster | Poster | New York City skyline at dusk. | 16.5 | West End Corp. |

Next 4 Matches

Clicking the **Next** button takes the user to the next page of results and displays a **Previous** button.



There are **8** matching records. Displaying matches **5** through **8**.

| ProdName | ProdType | ProdDesc | Price | Vendor Name |
|---|---|---|---|---|
| Oregon Coast Poster | Limited Edition Poster | Oregon coast photograph by Ian Johnson. Limited edition. | 12.5 | Worldwide Posters Unlimited |
| Brookings Coast Poster | Limited Edition Poster | Brookings Coast Poster by Ian Johnson. Limited edition. | 18.0 | Worldwide Posters Unlimited |
| Ghost Town Poster | Limited Edition Poster | Ghost town in winter. Limited edition photography by Lori Mathews. | 33.0 | Worldwide Posters Unlimited |
| Deer Poster | Limited Edition Poster | Deer peeking out behind a tree trunk. Limited edition photograph by Ian Johnson. | 12.0 | Worldwide Posters Unlimited |

Previous 4 Matches

# Formatting the Record List Page

Use the format options to define how the record list is displayed.

Tango displays results records in a table with one row for each record.

| First Name | Last Name | Department |
|---|---|---|
| Carolynn | Peterson | Sales |
| Dave | Heartsdale | Accounting |
| Linda | Stewart | Administration |
| Aaron | Smith | Accounting |
| Peter | Barken | Engineering |
| Linda | Jennings | Engineering |
| Peter | Jacobson | Sales |
| Richard | Frankin | Sales |
| Jenna | Smith | Administration |
| Erica | Manley | Sales |

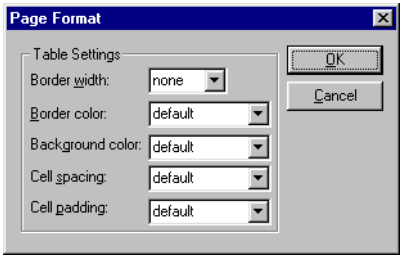
### *To change the format of the record list page*

Page Format

Do either of the following:

- In the Record List window, click the **Page Format** icon.
- In the main window, choose **Page Format** from the **Attributes** menu.

The Page Format dialog box appears.




Specify the table attributes as follows:

- **Border width**. The width of the table border in pixels. Select from numbers **1** to **8**. The default is **1**.

- **Border color**. The color of the table border. Select **default** or a color from the list.

**Note** For **Border color**, **Background color**, **Cell spacing**, and **Cell padding**, selecting **default** instead of a value omits that attribute from the HTML and uses the browser's default setting instead.

- **Background color**. The background of the table. Select **default** or a color from the list.
- **Cell spacing**. The amount of space, in pixels, inserted between individual cells in the table. Select **none**, or from numbers **1** to **8**.
- **Cell padding**. The amount of space, in pixels, between the border of a cell and the contents of the cell.

# Customizing Your Record List Page

## Header and Footer HTML

You use Header HTML and Footer HTML to customize the record list page by specifying HTML to appear above and below the record list.

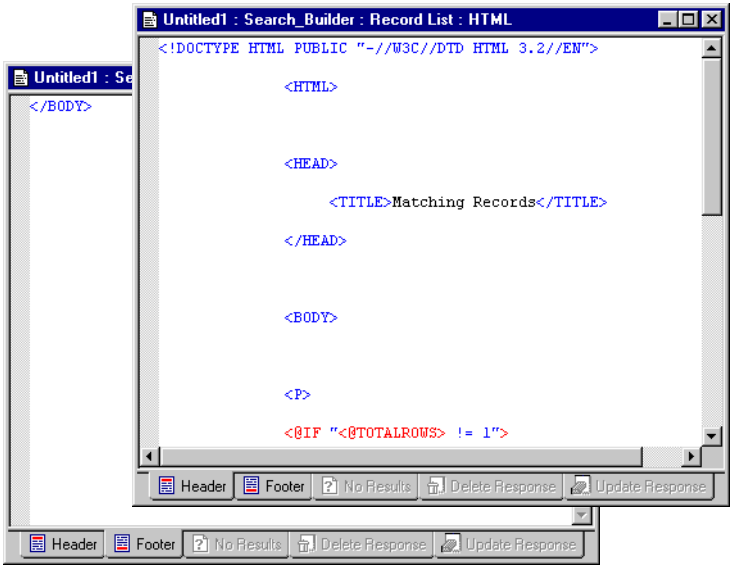
### *To enter Header HTML and Footer HTML*

1 Do either of the following:



- In the Record List window, click the **Header HTML** or **Footer HTML** icon.
- In the main window, choose **Header HTML** or **Footer HTML** from the **Attributes** menu.

The corresponding HTML editing window appears.



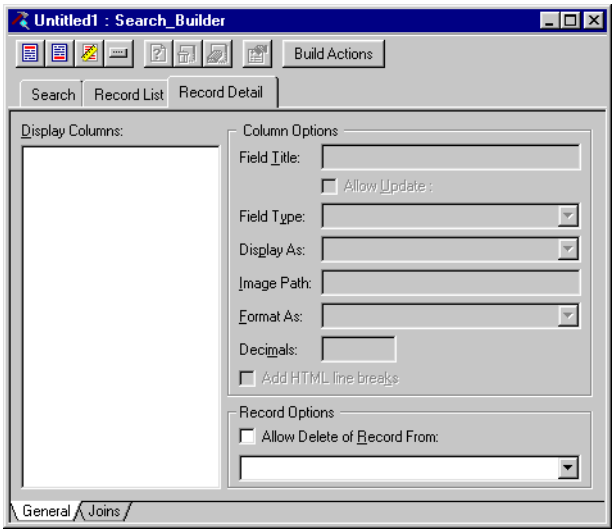

You can switch between the HTML editing windows by clicking on the **Header** and **Footer** tabs at the bottom of the HTML window.

2 Enter the desired HTML.

3 Close the editing window.

# Setting Record Detail Options

Use the options in the Record Detail window of the Search Builder to define the appearance and functionality of the Web page returned when a user clicks on a record on the record list Web page. This page displays a single record and supports user editing and deletion, if you choose to allow it.




## Display Columns

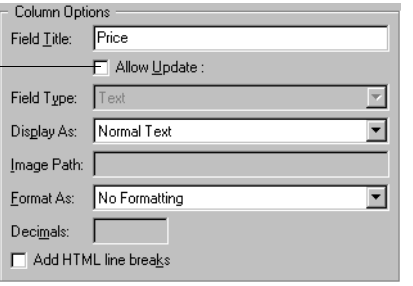
The columns appearing in this list are displayed on the record detail Web page. To add a column to the list, drag it from the Data Sources Workspace. The order in which columns appear in the **Display Columns** list is the order they appear on the record detail page.

## Column Options

Use the **Column Options** section of the Record Detail window to configure each detail column. This section describes each of the column options.

Other than the **Allow Update** option, the Column Options section for the Record Detail window is the same as the Column Options section for the Record List window. See page 167.

Allows the user to change the value of the column on the record detail page and save the changes to the database.




### Field Title

In the **Field Title** field, enter the title to appear for this column's value on the detail page.

### Allow Update

Select this option to allow the user to change the value of the column on the record detail page and save the changes to the database.

### Field Type

If you select the **Allow Update** option for a column, the **Field Type** drop-down list and **Field Properties** icon are enabled, allowing you to select the type of value editing field you want to appear for the column on the detail page.

As with the search form, you can select from the available field types: **Text**, **Drop-down List**, **List Box**, **Check Box**, and **Radio Buttons**.

For more information, see "Field Type" on page 157.

You specify the field type and its options the same way you do in the Search window of the Search Builder.

The selected column's Field Properties dialog box for each field type is the same in the Record Detail window as it is in the Search window, except you cannot specify a default value (text field type)

or a selected item (drop-down list, list box, check box, and radio buttons). This is because the value of the column in the detail record determines the field's initial value.

When creating value lists for drop-down list, list box, and radio button field types in the Record Detail window, make sure you enter the item values exactly as they appear in the database, and include all possible values. If Tango cannot find the column's value in the list when it is constructing the detail page for a record, no item is selected by default. Depending on the user's browser, the first item may be selected or no item may be selected. Either way, if the user saves the record—even if no changes are made to that particular field—a new value (an empty value or the first value in the list) is saved in it.

For similar reasons, make sure check box fields are used only for columns that can contain either an empty value or the value you specify as its checked value.

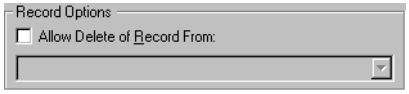
### Setting Column Options: Display As, Image Path, Format As, Decimals, and Add HTML line breaks

Setting these options is identical to setting the column options for the Record List page, except as follows:

- When you select the **Allow Update** option, these options are disabled.
- The **Display As** drop-down list excludes the **Link to Detail** option.

## Record Options

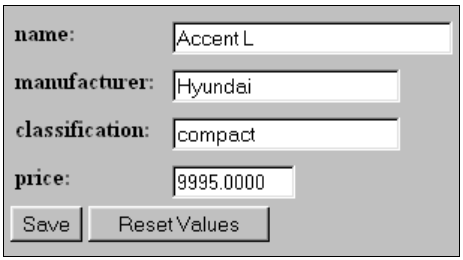
If you select the **Allow Delete of Record From** option, a **Delete** button is added to the record detail page, giving the user the ability to delete the current detail record.




Deleting records from multiple tables simultaneously is not supported by the Search Builder, so if you have included columns from more than one table in the **Display Columns** list, use the drop-down list to select the table whose record you want to delete.

# Formatting the Record Detail Page

Use the format options to define how the detail column values and their titles are displayed.



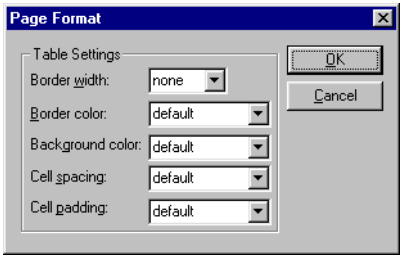

### *To change the format of the record detail page*

Do either of the following:


Page Format

- In the Record Details window, click the **Page Format** icon.
- In the main window, choose **Page Format** from the **Attributes** menu.

The Page Format dialog box appears.




Specify the table attributes as follows:

- **Border width**. The width of the table border in pixels. Select **none**, or from numbers **1** to **8**.
- **Border color**. The color of the table border. Select **default** or a color from the list.

**Note** For **Border color**, **Background color**, **Cell spacing**, and **Cell padding**, selecting **default** instead of a value omits that attribute from the HTML and uses the browser's default setting instead.

- **Background color**. The background of the table. Select **default** or a color from the list.
- **Cell spacing**. The amount of space, in pixels, inserted between individual cells in the table. Select **none**, or from numbers **1** to **8**.
- **Cell padding**. The amount of space, in pixels, between the border of a cell and the contents of the cell. Select **none**, or from numbers **1** to **8**.

# Customizing Your Record Detail Page and Creating Response Messages

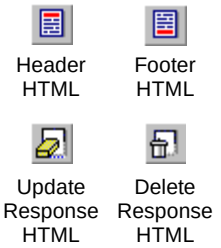
## Header, Footer, Update Response, and Delete Response HTML

You use Header HTML, Footer HTML, Update Response HTML, and Delete Response HTML to customize the detail page.

Using Header HTML and Footer HTML, you can edit the HTML that you want to appear above and below the record data.

Using Update Response HTML and Delete Response HTML, you create messages in response to record updates and deletions.

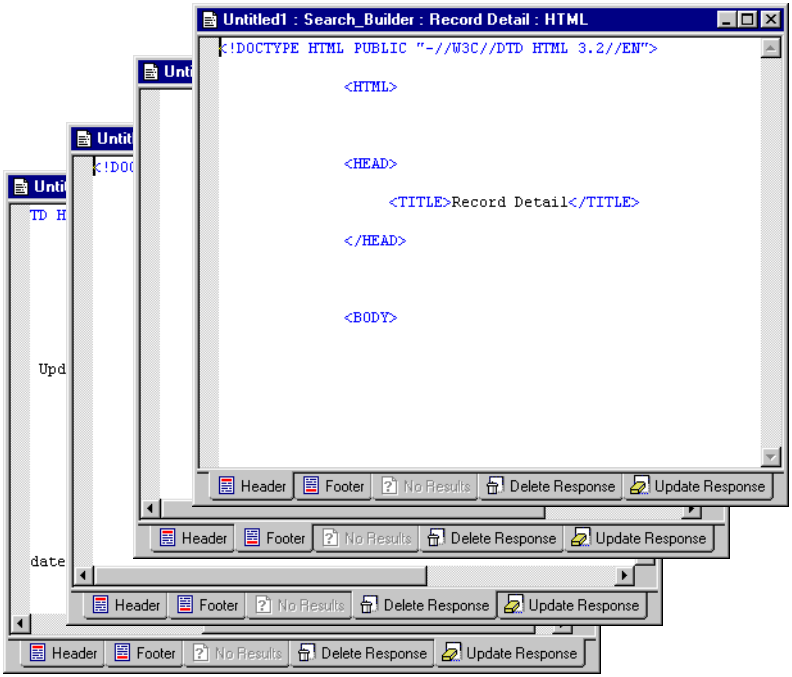
### *To enter Header HTML, Footer HTML, Update Response HTML, or Delete Response HTML*




1 Do either of the following:

- In the Record Detail window, click the **Header HTML**, **Footer HTML**, **Update Response HTML**, or **Delete Response HTML** icon.
- In the main window, from the **Attributes** menu, choose **Header HTML**, or **Footer HTML**, or choose **Responses** then **Update Response HTML** or **Delete Response HTML** from the submenu.

The corresponding HTML editing window appears.




You can switch between the HTML editing windows by clicking on the **Header**, **Footer**, **Delete Response**, and **Update Response** tabs at the bottom of the HTML window.

2 Enter the desired HTML.

3 Close the editing window.

## Button Titles

When you make a field updatable, or when you allow users to delete records from the detail page, buttons for these actions are added to the record detail Web page.

The record detail Web page contains three buttons below your record detail fields: **Save**, **Reset Values**, and **Delete**.

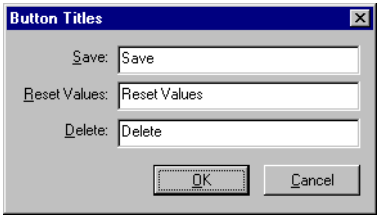
### *To change button titles*

Button Titles

1 Do either of the following:

- In the Record Detail window, click the **Button Titles** icon.
- In the main window, choose **Button Titles** from the **Attributes** menu.

The Button Titles dialog box appears.




2 Enter new titles in the corresponding fields.

3 Click **OK.**

# Search Builder Tips

You can use the Search Builder to create actions that do not follow the standard search form, record list, and record detail sequence.

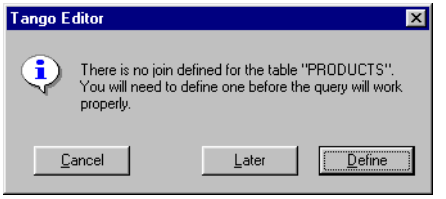
- **Using only fixed values in the Search options**. If you configure all of the Search columns to search for fixed values, the built actions on execution take the user directly to the record list and display the records matching the criteria you specify.
- **Specifying no Search columns**. If you do not specify Search columns, the built actions on execution take the user directly to the record list and display all the records in the database table.
- **Specifying no Record Detail columns**. If you do not specify Record Detail columns, the built actions do not contain detail functionality and no links appear in the record list.
- **Specifying no Record List columns**. If you do not specify Record List columns, the built actions on execution take the user straight to the detail page for the first record matching the Search criteria.

# Defining Joins

For complete details on what joins are and how to define them using Tango Editor, see “Joining Database Tables” on page 295.

You can include columns from more than one table in a search, if you define joins for the tables.

If you select columns from more than one table in a search, a dialog box appears telling you to define a join.



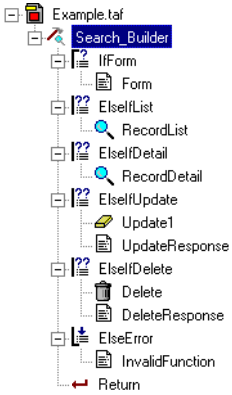

Either choose **Define** to go directly to the **Joins** tab or **Later** if you want to define the join at a later time.

When you define the join, it adds the columns to a search. You must, however, define the join before you build the actions for the search or you save the application file.

**Note** You must define separate joins for the initial search (the one that displays the record list) and for the detail search.

# Actions Built by the Search Builder

The actions built by the Search Builder appear in the application file as follows:




For more information, see “Building the Actions” on page 148.

The following table describes the actions resulting from the Search Builder process and the conditions under which actions are built:

### IfForm

`<@ARG _function> = form` or `<@ARG _function>` is empty

This section appears only if one or more user-enterable search columns are specified.

#### Form

### ElseIfList

`<@ARG _function> = list`

This section appears only if record list columns are specified. If no Form is present, this is an If action named IfList.

#### RecordList

If you selected **SQL Statement** for any column value, a Direct DBMS action (one for each) appears immediately before the RecordList action.

**ElseIfDetail**

```
<@ARG _function> = detail
```

This section appears only if detail columns are specified. If no RecordList or Form section exists, this is an If action named IfDetail.

**RecordDetail**

**ElseIfUpdate**

```
<@ARG _function> = update
```

This section appears only if updatable detail columns are specified.

**Update**

**UpdateResponse**

For information about the update response, see "Header, Footer, Update Response, and Delete Response HTML" on page 180.

**ElseIfDelete**

```
<@ARG _function> = delete
```

This section appears only if the **Delete** option is specified for the Record Detail page.

**Delete**

**DeleteResponse**

For information about the delete response, see "Header, Footer, Update Response, and Delete Response HTML" on page 180.

**ElseError**

**Invalid Function**

The HTML for this action displays the following message:

```
Error: Invalid Function
An unknown function was specified.
```

**Return**

## HTML Snippets

The Snippets Workspace contains a snippets folder named **Builder Snippets**, and a subfolder named **Search**. The **Search** folder contains snippets for the Form Header, Form Footer, Record List Header, Next / Previous Buttons, Record List Footer, No Matches, Record Detail Header, Record Detail Footer, Update Response, and Delete Response.

The Search Builder uses these snippets in the designated places as default values for the named attributes. To change the default values, you can edit these snippets.

CHAPTER TEN

# Configuring the New Record Builder

10

*Tango New Record Builder Options and Setup*

The *New Record Builder* builds a series of actions that allows users to add a record to a database. For the new record, you specify the database columns the user can add data to and the result message to return after the record is added. Tango does the rest.

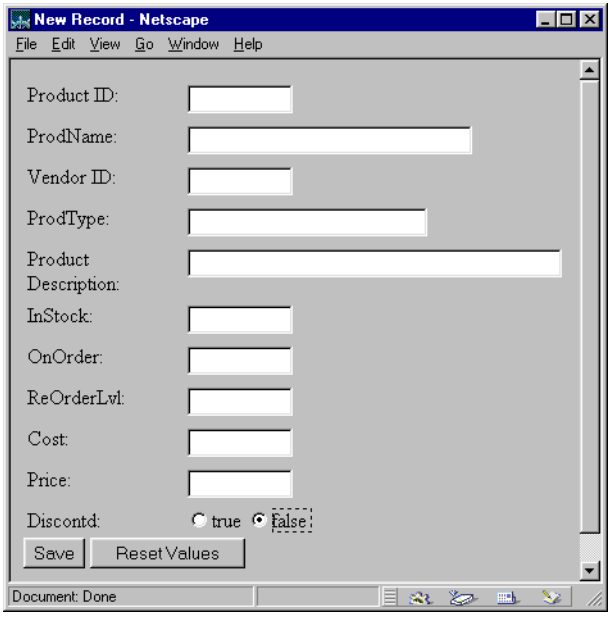
The topics covered in this chapter include:

- setting new record column options
- formatting the new record form
- customizing your form and creating result messages
- a summary of how to set column options.

# About the New Record Builder

You use the New Record Builder to build actions that, when Tango Server executes them, display a form allowing users to enter data for a new record and return a result message after the record is added to the database.

The following is an example of a new record entry form.



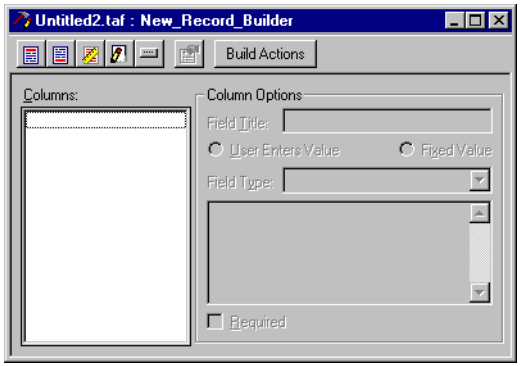

The user enters the values for the columns in the new record, and clicks **Save** to save the record. Tango Server saves the record to the data source and returns the HTML response you specified in the New Record Response HTML.


In this example, the response message has been customized to include values from the newly added record.

# Setting New Record Options

New Record Builder

When you drag the New Record Builder icon from the Actions bar into an application file, the New Record Builder window appears.




All the options necessary for configuring the New Record Builder appear in its options window.

## Columns

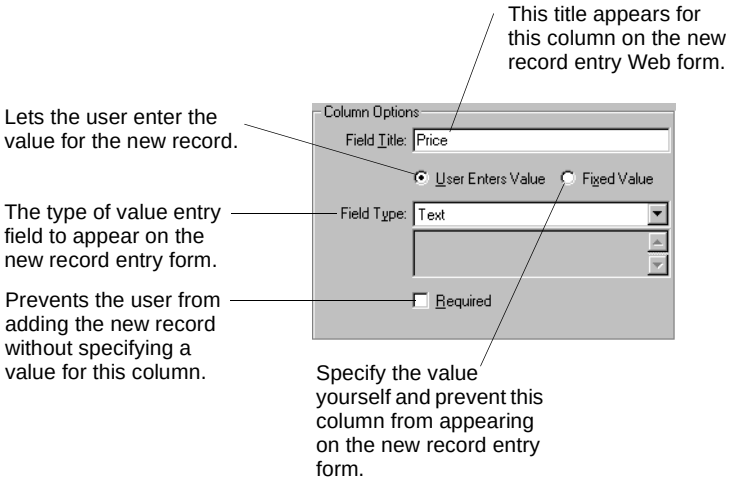
The columns you include in this list are the columns the user assigns values to in the new record. To add columns to the list, drag them from the Data Sources Workspace. Columns appear in the format *table_name.column_name*. You can only add columns from one table. The order in which the columns appear in the **Columns** list determines their order on the resulting new record entry form.

The following table describes the operations you can perform on columns.

| To ... | Do This ... |
|---|---|
| Reorder columns | Select the columns and drag them to a different location in the list. |
| Delete columns | Select the columns. Choose **Delete** from the **Edit** menu, press the DELETE key on the keyboard, or select the **Delete Selected Items** icon on the toolbar. |
| Delete columns without confirmation | Hold down the CTRL key while using the **Delete** command. |

## Columns Options

Use the **Column Options** area to configure each column appearing in the **Columns** list. You can specify how each column's entry field appears on the new record entry form and whether a value is required for it or not.




### Field Title

In this field, set the title of the value entry field for the column as you want it to appear on the new record entry form.

### User Enters Value

Select this option if you want the user to enter a value in an entry field on the new record entry form.

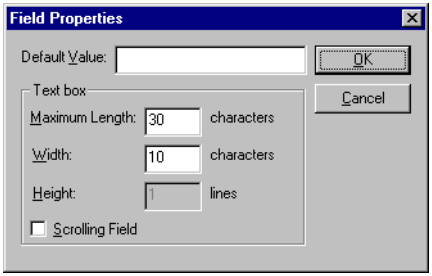
### Field Type

To select a value entry field type for a column, select the column in the **Columns** list and select an item from the **Field Type** drop-down list (**Text**, **Drop-down List**, **List Box**, **Check Box**, or **Radio Buttons**).

A Field Properties dialog box appears allowing you to specify properties for the field.

Field Properties

You can edit a field's properties at any time by clicking the **Field Properties** icon in the New Record Builder window or by choosing **Field Properties** from the **Attributes** menu. The Field Properties dialog box for the specified type of field appears. For example, in the case of a text field:




For more details on the different types of field that can appear here (text, drop-down list, list box, check box, or radio buttons), see "Field Type" on page 157.

The following describes each of the options in the Field Properties dialog box for each field type.

- **Default Value** is the default value you want to appear on the new record entry form.
- **Maximum Length** is the maximum number of characters the user can enter in the field. This option is not available when **Scrolling Field** is selected because HTML does not support it.
- **Width** is the width of the field in characters.
- **Height** is the number of lines of text displayed in the field without scrolling. This field is available only when you select **Scrolling Field**. The height of a non-scrolling field is always **1**.

## Fixed Value

If you select this option, no entry field appears on the new record entry form. The value you specify is used for every new record. Using the **Value** drop-down list, select one of the following options for a fixed value:

- **Value Entered**. Use the text box provided to enter the value for the **Field Title**.
- **SQL Expression**. The value returned by the SQL expression text entered is used as the value. The text entered is evaluated by the database and the result is used as the column value in the new record.

**Note** For ODBC data sources, you can enter ODBC scalar functions here.

- **SQL Statement**. The SQL statement entered is executed, the results are retrieved, and the first data item of the results is used as the column value in the new record. For example, if you enter:

  ```
  SELECT (MAX (cust_num)+1) FROM customer
  ```

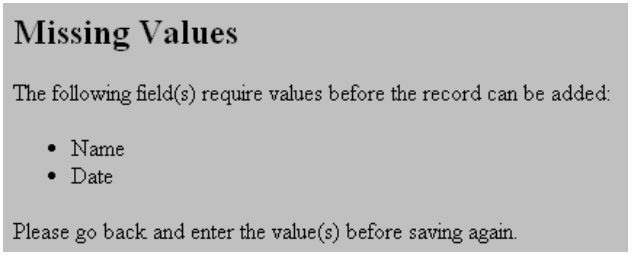
  the largest customer number plus one is used as the value for the column in the new record.

- **Current Timestamp**. The current timestamp (date and time combined) on the Tango Server computer is used as the value.
- **Current Date**. The current date on the Tango Server computer is used as the value.
- **Current Time**. The current time on the Tango Server computer is used as the value.
- **CGI Parameters**. The rest of the fixed value options are referred to collectively as CGI parameters. They include **Client Name**, **Client Domain**, **Client IP Address**, **Client Browser**, **Server Address**, **Server Port**, **Referer Page URL**, and **Method**. When you specify one of these parameters as the column value, the parameter value passed in when the user clicks the **Save** button is used.

For more information on what the CGI parameters evaluate to, see "Assigning Variables With the Assign Action" on page 130, and "<@CGIPARAM>" on page 50 of the *Meta Tags and Configuration Variables* manual.

### Required

Select this column option to prevent a record from being added, unless the user enters a value. If the user tries to leave the value field empty, an error message like the following is returned.




If you select **Fixed Value** for the column, the **Required** option is not available. It is also not available for columns configured to use the **Check Box** field type. This is because a check box can have only two values: empty and the one you specify in the Field Properties dialog box. Making it required would mean the record could not be

added unless the user checks the checkbox, in which case you could use the **Fixed Value** option for the column and have it not appear on the Web page.

## Summary: Setting Column Options

The following table summarizes how to set column options for the New Record Builder.

| To ... | Do This ... |
|---|---|
| Let the user specify the value for a column | Select the column in the **Columns** list. Select the **User Enters Value** option. This option defines the column as a user-enterable field, and a value entry field appears on the new record entry form. |
| Specify the title to appear for a column's new record form value entry field | Select the column in the **Columns** list. The column's name appears in the **Field Title** field. Replace the text with the desired field title. Tango remembers the entered title and uses it as the default the next time you choose it. |
| Specify the type of value entry field you want to display for a column | Select the column in the **Columns** list. Make sure the **User Enters Value** option is selected. From the **Field Type** drop-down list, select the type of field you want to display. A dialog box appears allowing you to specify the field attributes. |
| Prevent the user from omitting a column value | Select the column in the **Columns** list. Make sure the **User Enters Value** is selected. Select the **Required** option. If the user leaves the field empty and tries to save the record, an error message appears explaining the problem. |
| Hard-code the value for a column | Select the column in the **Columns** list. Select the **Fixed Value** option. From the **Value** drop-down list, select a value to use for the new record. You can use one of the preset values, such as **Current Date** or **Current Time**, or select **Value Entered** to enter a value yourself. Select one of the SQL options to get a value from the data source. Columns specified as Fixed Value do not appear on the new record entry form. |

# Formatting the New Record Form

Use the format options to determine the layout method to display the value entry fields and their titles.

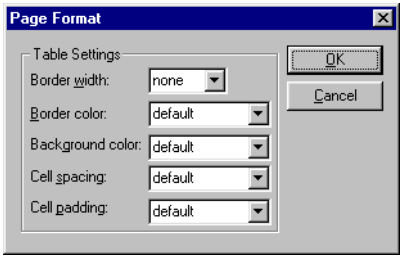
### *To change the page format options of the new record form*

Do either of the following:

Page Format

- In the New Record Builder window, click the **Page Format** icon.
- In the main window, choose **Page Format** from the **Attributes** menu.

The Page Format dialog box appears.




Tango aligns the page fields using an HTML table. Specify the table attributes as follows:

- **Border width**. The width of the table border in pixels. Select **none**, or from **1** to **8**. The default **none** has a value of zero.
- **Border color**. The color of the table border. Select **default** or a color from the list.

**Note** For **Border color**, **Background color**, **Cell spacing**, and **Cell padding**, selecting **default** instead of a value omits that attribute from the HTML (which tells the browser to use its default setting instead).

- **Background color**. The background of the table. Select **default** or a color from the list.
- **Cell spacing**. The amount of space, in pixels, inserted between individual cells in the table. Select **none**, or from **1** to **8**.
- **Cell padding**. The amount of space, in pixels, between the border of a cell and its contents. Select **none**, or from **1** to **8**.

# Customizing Your Form and Creating Result Messages

The Snippets Workspace includes the default builder HTML snippets described in this section. They include snippets for Form Header, Form Footer, and New Record Response.

## Header, Footer, and New Record Response HTML

You use Header HTML and Footer HTML to customize the new record entry form by specifying HTML that is placed above and below the entry form.

The New Record Response HTML is returned after the user saves the new record.

The Column Snippets folder in the Snippets Workspace contains the names of all the columns in the table being inserted into. Dragging a column name to the HTML editing field causes an `<@COLUMN>` tag to be added to the HTML. When the application file is executed, the column's value in the new record is included at that location in the response.

To display values from the new record, Tango must be able to get those values from one of three places: the new record form submitted by the user, a fixed value you have specified, or from the database. If you include a column in the result message that does not appear in the **Columns** list of the New Record Builder window, Tango must do a search of the database to retrieve the new record after it is added.

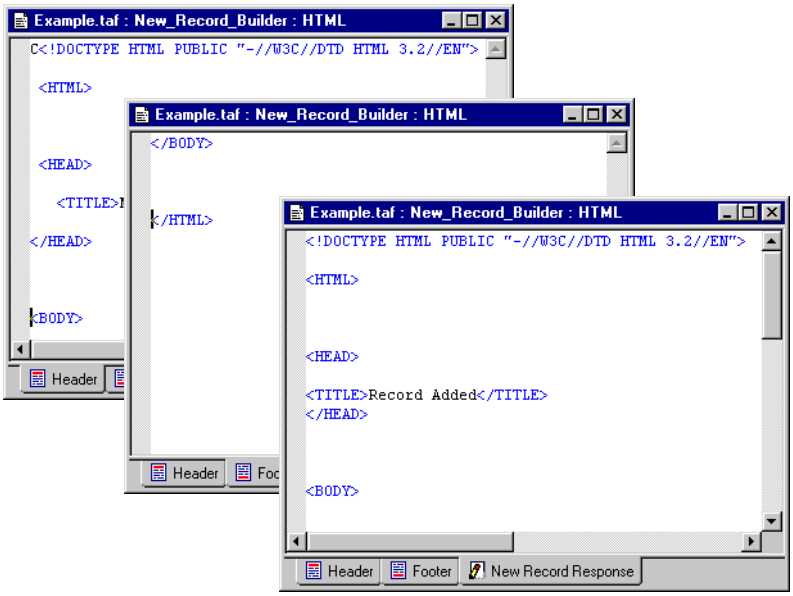
To do so, Tango must have the value of the record's primary key column(s). This means the primary key column(s) must appear in the **Columns** list of the builder. Without this information, Tango Editor does not permit you to build the New Record actions.

### *To enter Header HTML, Footer HTML, or New Record Response HTML*


**1** Do either of the following:

- In the New Record Builder window, click the **Header HTML**, **Footer HTML** or **New Record Response HTML** icon.
- In the main window, choose **Header HTML**, **Footer HTML**, or **New Record Response HTML** from the **Attributes** menu.

The corresponding HTML editing window appears.




**2** If you want to include HTML other than the default snippets, enter it.

**3** Close the editing window.

## Changing Button Titles

The new record entry form contains two buttons at the bottom of your form:

- The **Save** button saves the record.
- The **Reset Values** button resets the entry fields to their default values.

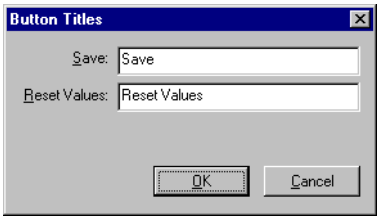
### *To change button titles*

**1** Do either of the following:

Button Titles

- In the New Record Builder window, click the **Button Titles** icon.
- In the main window, choose **Button Titles** from the **Attributes** menu.

The Button Titles dialog box appears.

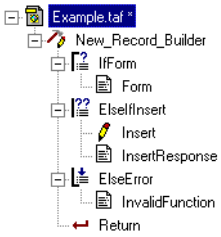



**2** Enter new titles in the corresponding fields.

**3** Click **OK.**

## Actions Built by the New Record Builder

The actions built by the New Record Builder appear in the application file window as follows:



For more information, see “Building the Actions” on page 148.

The following shows the actions resulting from the New Record Builder process and the conditions under which the actions are built:

**IfForm**

```
<@ARG _function> = form
```

This section appears only if one or more user-enterable fields exist.

**Form**

**ElseIfInsert**

```
<@ARG _function> = insert
```

If no form is present, this is an If action named IfInsert.

**IfMissingRequiredFields**

This action contains one criterion for each required field, checking to see if <@ARG *fieldName*> is empty. All the criteria are OR’d.

If there are no required fields, this If/Else condition does not exist.

**MissingFieldsMessage**

If the user leaves out values for required fields, the HTML for this action displays the following message:

```
Error: Missing Required Fields

The record could not be added because the
following required fields were left empty:

...

Please go back and enter values for these
fields.
```

**ElseDoInsert**

**Insert**

If you selected **SQL Statement** for any column value, a Direct DBMS action (one for each) appears immediately before the Insert action.

**InsertResponse**

For information about the new record entry response, see “Header, Footer, and New Record Response HTML” on page 197.

**ElseError**

**Invalid Function**

The HTML for this action displays the following message:

```
Error: Invalid Function
An unknown function was specified.
```

**Return**

## HTML Snippets

The Snippets Workspace contains a snippets folder named **Builder Snippets**, and a subfolder named **New Record**. The New Record folder contains snippets for the Form Header, Form Footer, and New Record Response.

The New Record Builder uses these snippets in the designated places as default values for the named attributes.

CHAPTER ELEVEN

# Using Actions

11

## The Basics of Using Tango Actions

A Tango application file is made up of a series of one or more *actions*. Each action performs a specific type of function and can have results, usually in the form of HTML*, associated with it. The applications you create may be used to input data to information systems, compose and display information from data sources, and many more interactions.

When an application file is called, the actions in it are executed by Tango Server. When execution is complete, the HTML results are returned to the user's browser. These results can be from the user or from interaction with other servers, normally DBMSs.

Several actions allow you to search, add, update and delete database records. There are also actions for executing manually entered database statements and controlling the flow of execution within an application file. You can also automatically create a sequence of actions using the builders.

This chapter covers the following topics:

- the Actions bar
- working with actions
- assigning attributes to actions.

* Tango does not restrict its content to only HTML format. Using other markup languages such as SGML, VRML, and XML instead of HTML is also possible.

# About Actions

The Actions bar shows all the available action types. It appears whenever an application file is active, or you choose **Actions Bar** from the **View** menu.


**Tip** You can drag the Actions bar to anywhere on your desktop and resize it.

You add all Tango actions to an application file from the Actions bar.

The following table lists each action, its function, and where in this *User's Guide* you can find more information.

| Icon | Action | Function | User's Guide Reference |
|---|---|---|---|
| | Select | Selects actions in the application file window. | This chapter. |
| | Search | Retrieves records from a database. | "Searching a Database" on page 228 |
| | Insert | Adds records to a database. | "Adding Records to a Database" on page 242 |
| | Update | Changes records in a database. | "Modifying a Database Record" on page 243 |
| | Delete | Removes records from a database. | "Removing a Database Record" on page 245 |
| | Direct DBMS | Executes SQL statements. | "Executing SQL" on page 290 |
| | Begin Transaction, End Transaction | Begins a transaction and ends a transaction with a rollback or commit. | "Creating Database Transactions" on page 286 |

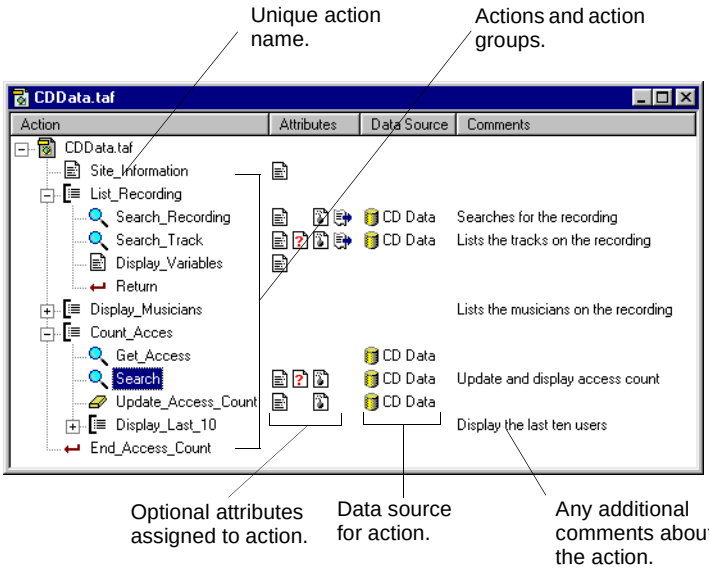
| Icon | Action | Function | User's Guide Reference |
|---|---|---|---|
|  | Results | Performs no special functions of its own, but it lets you append HTML to the results. | This chapter. |
|  | Mail | Sends out electronic mail. | "Sending Electronic Mail From Tango" on page 275 |
|  | File | Reads, writes, and deletes files on the Tango Server machine. | "Reading, Writing, and Deleting Files" on page 279 |
|  | Script | Allows you to specify server-side JavaScript code to execute. | "Executing JavaScript" on page 264 |
|  | External | Calls an external code module to perform a function and return results. | "Using an External Action" on page 266 |
|  | Assign | Makes specified value assignments. | "Assigning Variables With the Assign Action" on page 130 |
|  | Group | Groups related actions. | "Grouping Actions" on page 221 |
|  | If, Else If, Else | Executes an expression and based on the result of the expression affects the control flow in the application file. | "Conditional Action Execution (If Action)" on page 252 |
|  | While Loop, For Loop | Repeats a set of contained actions: until an expression evaluates to true or for a specified number of times. | "Repeating Actions (Loop Actions)" on page 257 |
|  | Break | Terminates processing in a loop. | "Exiting a Loop (Break Action)" on page 261 |
|  | Branch | Causes a jump to another action or action group. | "Jumping to a Designated Action (Branch Action)" on page 248 |
|  | Return | Ends execution of the application file and returns the accumulated Results HTML to the Web browser. | "Ending File Processing (Return Action)" on page 262 |

As well as actions, the Actions bar includes icons for the Search Builder and the New Record Builder. You add the builders to an application file in exactly the same way you add actions.

| Icon | Builder | Function | User's Guide Reference |
|---|---|---|---|
|  | Search Builder | Builds the actions required to perform a search. | "Configuring the Search Builder" on page 151 |
|  | New Record Builder | Builds the actions required to add a new record. | "Configuring the New Record Builder" on page 189 |

# Working With Actions

The application file window shows the actions that you want Tango Server to execute. Generally speaking, actions are executed sequentially, from top to bottom, until a control action is encountered. Control actions make decisions and cause execution to jump to another action or action group.




An icon indicates the type of action, and each action must have a name.

**Note** Each action name in the application file must be unique.

An action can also have attributes. Once assigned, action attribute icons appear beside the action name in the Attributes column indicating which actions have what attributes associated with them.

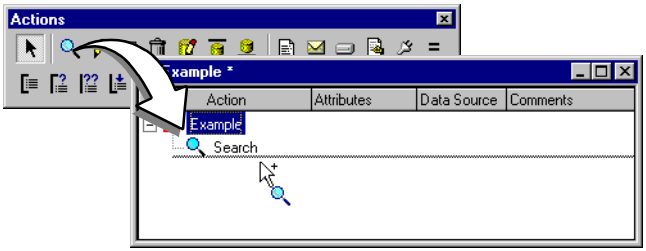
Database operations in the application file operate on the data source each is assigned to.

## Adding an Action

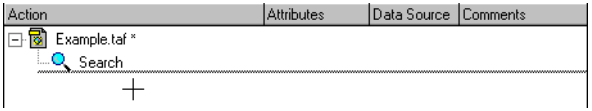
### *To add an action to an application file*

Do either of the following:

- Drag an action icon from the Actions bar into the application file window (the cursor changes to include crosshairs and the action icon you are adding), and drop it where you want to add the action.




- Click an action icon, move the cursor into the application file window (the cursor changes to crosshairs), and click where you want to add the action.




In either method, a gray line indicates where the new action is to be placed.

If the action has an editing window, it opens automatically.

**Tip** To prevent the action's editing window from being opened automatically, hold down the CTRL key while dragging the new action into the document window.

## Naming an Action

Each action in an application file must have a unique name. Tango Editor gives actions a unique name automatically.

The default name for an action is its action type. When you add an action that already exists in the application file with its default name, Tango appends the default name with a numeric starting at "1", for example, "Search1"

**Tip** To make your application files more readable, you should always replace default names with more meaningful ones.

### *To change the name of an action in an application file*

1. Select the action you want to rename.
2. Click the action's name, or from the **Edit** menu, choose **Rename**.
3. Type the new name.

You can also right click the name, and from the menu that appears, click **Rename**.

**Note** Action names can contain only letters, numbers, and underscores. No spaces, punctuation, or other characters are allowed. Adding spaces automatically adds underscores.

When you change the name of an action, Tango automatically updates any Branch actions in the same application file referring to the action. If you change the name of an action that is the destination for branches from other application files, the Branch actions in other application files are *not* updated.

Tango does *not* automatically update action results references for renamed actions.

## Deleting an Action

### *To delete an action from an application file*

1. Select the action you want to delete.
2. Do one of the following:
   - On the main toolbar, click the Delete icon.
   - From the **Edit** menu, choose **Delete**.
   - Press the DELETE key.

3 When the dialog box appears asking you to confirm the deletion, choose **OK**.

**Tip** You can bypass the confirmation dialog box by holding down the CTRL key when choosing **Delete**.

## Editing an Action

All of the actions—except Return, Group and Break actions—have associated attributes and parameters. You can set these parameters in the action's editing window.

### *To edit an action in an application file*

Double click the action icon in the application file window.

The action's editing window opens.

If the action is associated with a data source, the Data Sources Workspace opens, listing the tables and columns for the data source. If Tango Editor has not loaded the data source yet, it is loaded first.

## Moving an Action

Tango executes the actions in an application file sequentially, from top to bottom; however, you can use control actions to modify this sequence.

If you want the actions to be performed in a different order, you can rearrange them. Move them to another location in the application file by dragging them to the position you want.

### *To move an action to a new location*

Do either of the following:

- Select the action you want to move, and drag the action to its new position.
- Select the action, and cut and paste it using the edit commands.

  Actions are pasted after the currently selected action, or at the end of the file if no action is selected.

  Edit commands are available from the Tango Editor **Edit** menu and from the context-sensitive menu.

When you move an action, Branch actions referring to it continue to branch to the action, even though its position has changed.

## Copying an Action

You may want to create an action that performs a task similar to one performed by an existing action in another application file. Instead of having to recreate the action and specify all its parameters again, Tango Editor allows you to duplicate an action.

### *To copy an action in the same application file*

Do either of the following:

- Select the action you want to copy, and hold down the CTRL key and drag the action to where you want the new action to appear.
- Select the action, and copy and paste it using the edit commands.

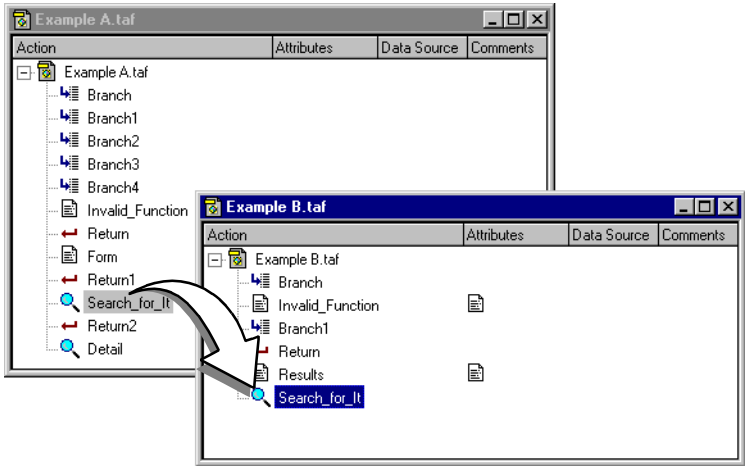
  Edit commands are available from the Tango Editor **Edit** menu and from the context-sensitive menu.

The copied action is given a new, unique name, which you should change to a more descriptive name.

### *To copy an action into another application file*

Do either of the following:

- Select the action you want to copy, and drag the action into another application file.




- Select the action, and copy and paste it using the edit commands.

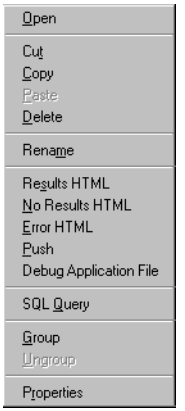
  Edit commands are available from the Tango Editor **Edit** menu and from the context-sensitive menu.

Be careful when copying database actions. For an action to work correctly in the new application file, the data source must be the same as in the original one.

Alternatively, you may assign another data source to the action in the new application file.

## Context-Sensitive Action Menu

When you right click an action icon in the application file window, or anywhere in the file window with an action selected, a context-sensitive menu of action commands appears.



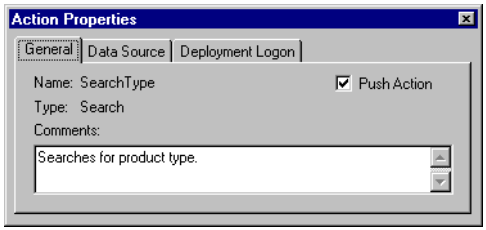

- **Open** opens the action editing window for the selected action.
- **Cut**, **Copy**, **Paste** and **Delete** perform the standard window editing functions.
- **Rename** allows you to edit the current name of the action.
- **Results HTML**, **No Results HTML**, **Error HTML**, and **Push** are attributes you can assign to actions which support them.
- **Debug Application File** is an attribute of the entire application file.
- **SQL Query** opens the SQL Query window so you can perform SQL queries from within Tango.
- **Group** and **Ungroup** allows you to group related actions and also to ungroup them.
- **Properties** displays the action properties window.

For more information on using these commands, see: "Assigning Attributes to Actions" on page 214, "Debugging Application Files" on page 44, "The SQL Query Window" on page 35, "Grouping Actions" on page 221, and "Action Properties" on page 213

## Action Properties

When you select an action and choose **Properties** from either the **Edit** menu or the context-sensitive menu, the Action Properties window for that action appears.

This window displays current information about the selected action and the assigned data source.




Using this window, you can change some of the action's properties.

For more information, see "Properties Window" on page 23.

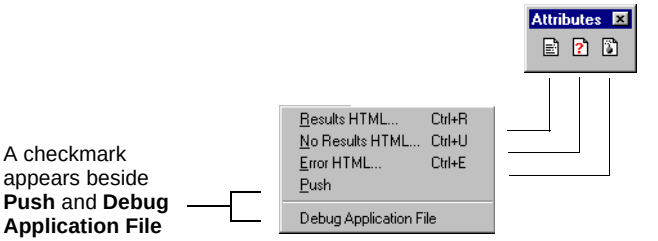
# Assigning Attributes to Actions

In addition to the parameters specific to each action type, which are edited using the action's editing window, actions can also have the following attributes:

- **Results HTML** applies to all actions, except control actions (other than Branch). After the action is executed, this HTML is added to the results returned.
- **No Results HTML** applies only to Search, Direct DBMS, Script, File, and External actions. When the action does not return data, this HTML is returned instead of the Results HTML.
- **Error HTML** applies to most action types except certain control actions (including Return and Break). In the event of an error in the action's execution, this HTML is returned immediately.
- **Push** causes the Results HTML accumulated so far to be sent back to the Web browser when the action to which it is assigned finishes executing. Execution then continues normally.
- **Debug Application File** lets you see useful information about your application file execution in your Web browser application. This attribute applies to the entire application file, not a particular action.

To assign any of these attributes to an action (or, for **Debug Application File**, to the application file), select or open the action editing window, then select an attribute from the **Attributes** menu or from the Attributes bar.

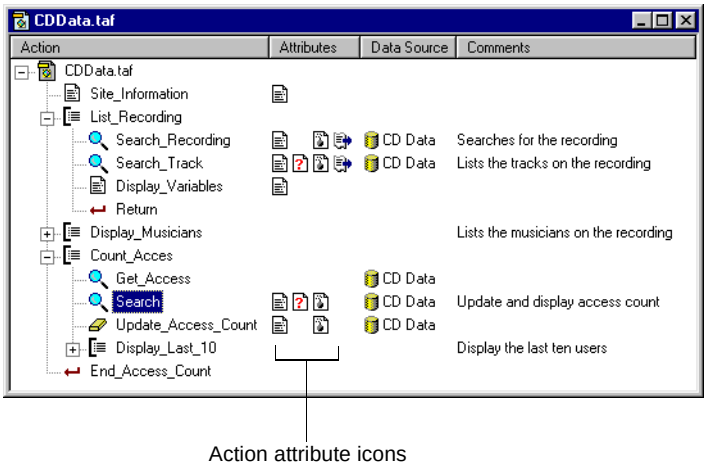
You can also right click the application file in Tango Editor and choose the attribute(s) that apply to that action from the context sensitive menu that appears.




A checkmark appears beside **Push** and **Debug Application File** when they are selected.

The HTML action attributes in the **Attributes** menu have a corresponding button on the Attributes bar.

Action attribute icons appear beside the action name in the Attributes column of the application file window to indicate which actions have any of these attributes associated with them.




For more information, see “HTML Editing Window” on page 24.

You can switch between the Results HTML, No Results HTML and Error HTML associated with an action by clicking on the tabs at the bottom of the HTML editing window.

## Results HTML

Many actions in an application file can have HTML associated with them. This HTML is stored in the Results HTML attribute. If Results HTML contains any text, the Results HTML icon appears in the attributes column of the application file window; otherwise, it does not.

As Tango Server executes the actions in a file, the Results HTML associated with each is accumulated. When execution of the file is complete, the HTML is returned.

Results HTML can also contain Tango *meta tags* that Tango Server processes. While all the other text in Results HTML is interpreted by the user’s browser and returned as is (via the Web server), Tango Server first substitutes meta tags with other values.

For more information, see “Using Meta Tags” on page 109.

The `<@COLUMN>` meta tag causes a database value to be placed in the HTML. There are many others, including tags for referencing form field and search argument values, and conditional tags for displaying HTML only if the result of a given comparison is true.

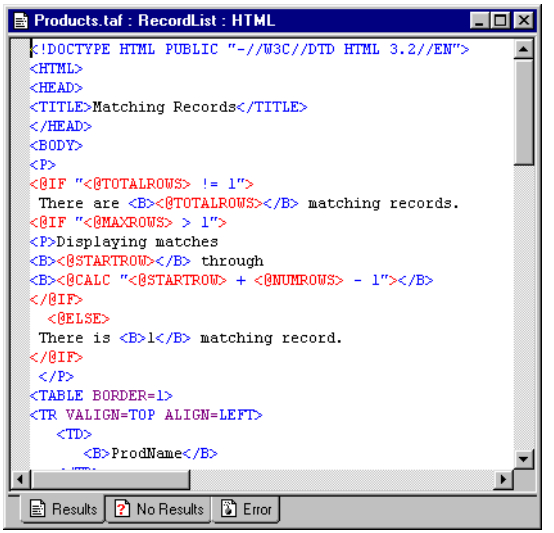
### To edit the Results HTML for an action

1 Select the action in the application file window.

2 From the **Attributes** menu, select **Results HTML**, or click on the Attributes bar.

You can also right click the action and choose **Results HTML** from the context-sensitive menu that appears.

The Results HTML editing window appears.




3 Type the Results HTML into the HTML text area. The text can include any valid HTML* or Tango meta tags.

You can switch between the Results HTML, No Results HTML and Error HTML associated with an action by clicking on the tabs at the bottom of the HTML editing window.

For more information, see “Working With Snippets” on page 101.

You can add column values (for Search actions only) and any HTML snippets you have defined to the Results HTML editing window from the Snippets Workspace. As well, you can add from the list of standard Tango snippets that allow for easy entry of many of the meta tags.

To include any of these items in your Results HTML, select the snippet and either drag it, or copy and paste it into the desired location in your text.

---

* Tango does not restrict its content to only HTML format. Using other markup languages such as SGML, VRML, and XML instead of HTML is also possible. If you use other content types, you are responsible for setting the HTTP header appropriately.

For HTML snippets that have placeholders for the current selection, select the text and drag the snippet over the selected text. The snippet is wrapped around the selection. For example, "`Title`" becomes "`<H1>Title</H1>`".

You can also easily add many of the common Tango meta tags.

1. Click the editing area where you want to add a meta tag.
2. Do either of the following:
   - From the **Edit** menu, choose **Insert Meta Tag**.
   - Right click, and from the context-sensitive menu that appears, choose **Insert Meta Tag**.

   The Insert Meta Tag dialog box appears.

For information on using the Insert Meta Tag dialog box, see "To insert common meta tags into your application file" on page 113.

## No Results HTML

You can associate No Results HTML text with Search, Direct DBMS, Script, and External actions. If the action execution does not return any data, this text is added to the application file's accumulated HTML instead of the Results HTML. This is useful when you want to display a special message to users when their queries do not return data.

**Note** If both Results HTML and No Results HTML appear as attributes, Tango accumulates one or the other, but never both.

After Tango Server processes the No Results HTML, execution of the application file continues normally to the next action.

No Results HTML can contain any of the Tango meta tags used in Results HTML, except for those related to displaying result data items, such as `<@ROWS>`, `<@COLUMN>`, and `<@COL>`.

## Error HTML

Error HTML allows you to specify your own error messages in HTML format, instead of having Tango Server produce them. The other alternative is to modify the `Error.htx` file; see "To specify your own custom default error message" on page 218.

You can associate Error HTML with most actions. If an action fails for any reason, execution ends and the Error HTML for the action is returned immediately to the user.

Error HTML can contain all the Tango meta tags used in Results HTML, except for those related to displaying result data items.

For more information, see "<@ERROR>" on page 79 and "<@ERRORS> </@ERRORS>" on page 81 of the *Meta Tags and Configuration Variables* manual.

There are also special Tango meta tags for displaying error information.

If no Error HTML has been assigned to an action and an error occurs in that action, Tango returns a default error message using the following HTML:

```
<h3>Error</h3>

An error occurred while processing your request:<p>
<@ERRORS>
Position: <b><@ERROR PART=POSITION></b><br>
Class: <b><@ERROR PART=CLASS></b><br>
Main Error Number: <b><@ERROR PART=NUMBER1></b><br>
<@ifequal <@ERROR PART=NUMBER2> 0>
<@else>
       Secondary Error Number: <b><@ERROR
PART=NUMBER2>
</b><br>
</@ifequal><p>
<i>
<@ERROR PART=MESSAGE1><br>
<@ifequal @ERROR PART=MESSAGE2> "">
<@else>
       @ERROR PART=MESSAGE2><br>
</@ifequal><p>
</i>
</@ERRORS>
```

### *To specify your own custom default error message*

1. Create a text file containing the desired HTML and meta tags.
2. Name the file `error.htx`.
3. Place it in the `C:\WinNT\Tango3\` directory under Windows NT, or in the `C:\Windows\Tango3\` directory under Windows 95.

If Tango Server finds this file, it processes and returns it instead of the built-in default Error HTML. Error HTML assigned to an action is used if it exists.

For more information, see "Configuring Tango Server" on page 317.

The name and location of this file is determined by the `defaultErrorFile` configuration variable, which can be modified using the `config.taf` application file. The values when Tango is first started are given above. If you modify the path or name of the error file, place the file in the directory you specified instead.

## Push

The **Push** attribute causes the Results HTML accumulated so far to be sent back to the Web browser, when the action to which the **Push** attribute is assigned finishes executing. Execution then continues.

Normally, Tango waits until all execution is finished before returning the results at one time. If you want the user to see some of the results while Tango continues with the rest of the execution, set the **Push** attribute of the action.

**Note** Some browsers may not display table HTML immediately if you use the push attribute to return an unclosed table.

## Debug Application File

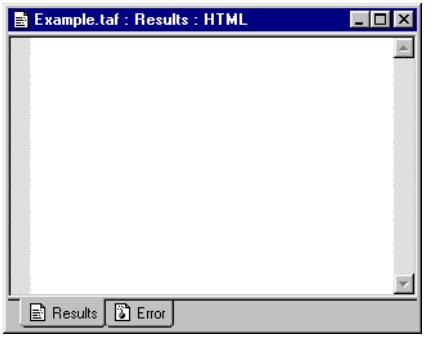
For more information, see “Debugging Application Files” on page 44.

# Adding HTML (Results Action)

Results Action

The Results action adds HTML to an application file's results.

When you drag the Results action icon from the Actions bar into an application file, a blank HTML editing window appears.




Results HTML can contain special meta tags that Tango Server processes. While all the other text in Results HTML is returned as is to your Web browser (via the Web server), any meta tags are first substituted with other values by Tango Server. You can also associate Error HTML with the Results action.

CHAPTER TWELVE

# *Grouping Actions*

12

*How to Organize Related Actions in an Application File*

The *Group* action allows you to organize actions within the file by grouping related actions and naming the group.

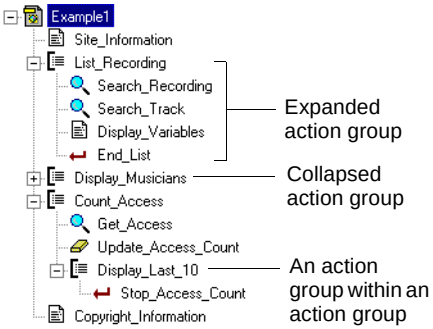
The topics in this chapter cover:

- working with action groups
- executing grouped actions.

# About Grouped Actions

In large Tango application files, it is common to place a number of related actions together; however, when viewing an application file it can be difficult to locate the related actions. To allow you to organize actions within the file better, you can group related actions and name the group. They also provide a destination for branching.

The following shows a typical application file containing action groups.




For more information, see “Application File Window” on page 40.

You can view grouped actions in a collapsed or expanded state. Groups added to an application file are in the expanded state by default. The collapsed or expanded state of an action group is saved in the application file. When the file is next opened, the last state is restored.

You can also include an action group within another action group.

You cannot associate action attributes with an action group. When you select a group, Results HTML, No Results HTML, Error HTML and Push attributes are disabled.

# Working With Action Groups

## Adding an Action Group

### *To add an action group to an application file*

Do either of the following:


Group

- From the Actions bar, drag the Group icon to anywhere in the application file.
- Select the actions you want to group together, and choose **Group** from the **Edit** menu or the context-sensitive menu.

  A new action group containing all selected actions is created and positioned where the top-most selected action was.

**Note** You can select discontiguous actions in the application file and drag them into a group; the actions do not need to be together already.

For more information, see "Naming an Action" on page 208.

Once added, the action group appears with a default name, just like other actions. You rename action groups in the same way you rename other actions. Just like action names, the action group name must be unique within the application file.

## Adding an Action to a Group

### *To add an action to a group*

To add an action to a group, do one of the following.

- Drag an action between two actions that are already in the group.

  The action is added to the group at that location.

- Drag an action onto the group icon.

  The action is added to the bottom of the group.

- Use the **Copy** and **Paste** commands to copy and paste an action into a group.

## Removing an Action From a Group

***To remove an action from a group***

Drag the action you want to remove outside the group.

If you drag the action above or below the group, the action appears immediately before or after the group, respectively.

**Note** Removing actions from a group does not delete the group, even if *all* actions are removed from the group.

## Ungrouping Actions

***To ungroup all actions with a group***

Do either of the following:

- Select the Group action.
- Choose **Ungroup** from the **Edit** menu or from the context-sensitive menu.

This deletes the group action but keeps the actions that were within the group.

## Deleting an Action Group

You delete an action group and all actions within it the same way you delete any action.

For more information, see “Deleting an Action” on page 209.

## Effect on Actions of Editing an Action Group

Moving an action group automatically moves all actions within the group with it.

Copying an action group copies all actions within the group as well.

Deleting an action group deletes the action group and all actions within the group.

## Branching to an Action Group

You can specify an action group as the destination of a Branch action.

For more information, see “Jumping to a Designated Action (Branch Action)” on page 248.

# Executing Grouped Actions

When Tango Server encounters an action group during file execution, no operation is performed on the action group itself, only on the actions within the group.

For more information, see “Exiting a Loop (Break Action)” on page 261.

Even though an action group has no effect on the execution of an application file, Tango Server supports the ability to branch to an action group and the ability to break out of an action group. If a Break action is encountered within a group, the next statement to be executed is the first statement outside of the group.

CHAPTER THIRTEEN

# 13 Using Basic Database Actions

*Setup and Operation of Search, Insert, Update, and Delete Actions*

Tango includes several fundamental database actions that allow you to search (*Search* action), add (*Insert* action), modify (*Update* action), and delete (*Delete* action) database records.

The topics covered in this chapter include setting up and executing Search, Insert, Update, and Delete actions.

# Searching a Database

Search actions retrieve database records matching a given criteria.

You use the Search action editing window to define what columns are selected, the order of the data retrieved, and the criteria that determine which rows are found.

**Tip** The SQL Query window gives you a convenient way to look at your database values. Choose **SQL Query** from the **Windows** menu or from the context-sensitive menu that appears when you right click the Search action editing window. For more information, see "The SQL Query Window" on page 35.

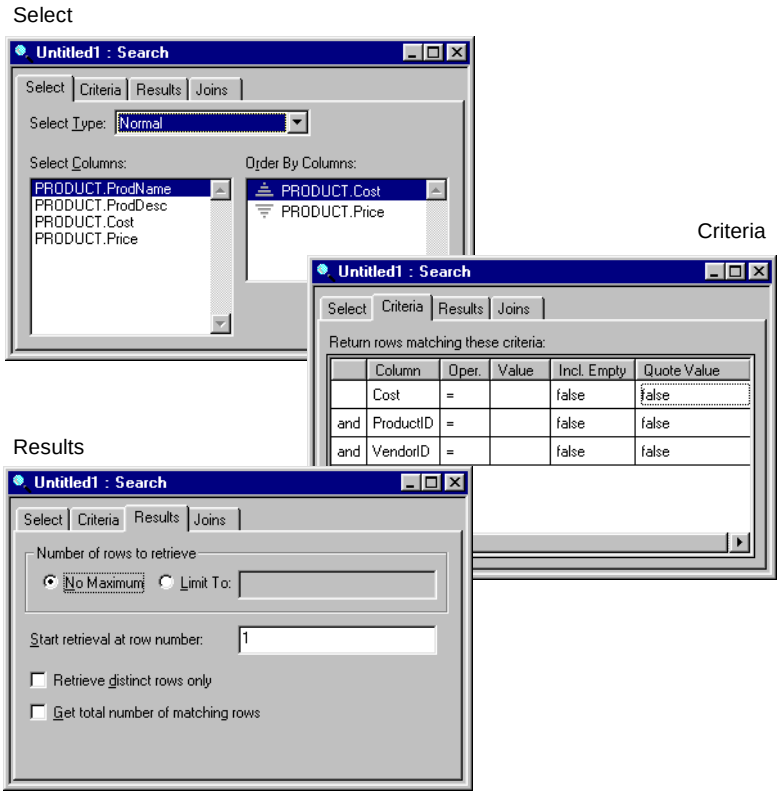
You use the action's Results HTML to format the results of the search.

## Setting Up a Search Action

Search Action

When you drag the Search action icon from the Actions bar into an application file, the search action editing window appears. The window contains tabs for the three main groups of settings for a Search action: Select, Criteria, and Results.

For details on the Joins tab, see "Working With Joins" on page 297.

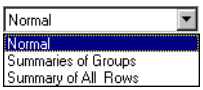



## Select Tab

You use the Select options to select the type of search to perform, the columns to retrieve, and the ordering of the records returned.

You can perform three main types of searches with a Search action: **Normal**, **Summaries of Groups**, and **Summary of All Rows**.

Select which type of search you want to perform from the **Select Type** drop-down list.



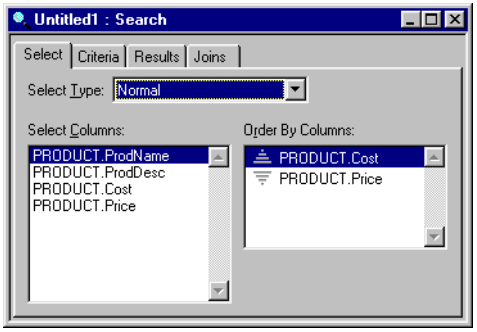

- **Normal** returns rows matching specified criteria.
- **Summaries of Groups** returns summaries of rows whose values in given columns (the grouping columns) are the same.

- **Summary of All Rows** returns a single row summarizing all rows matching your criteria. This kind of search lets you get information such as the maximum or average value of a particular column in a database table.

### *Normal Search*

The **Normal** type of search returns rows matching specified criteria. This is the most common type of search.

When you select **Normal** from the **Select Type** drop-down list, the Search action's Select window appears.




Specify values for the parameters in the Select window as follows:

- **Select Columns**. Drag into this list from the Data Sources Workspace the columns whose data is to be retrieved from the database.

  You can include columns from multiple tables; if you do, you must define joins to describe how the tables are related.

For more information, see "Joining Database Tables" on page 295.

- **Order By Columns**. Drag into this list the columns that are used to sort the results returned to the user. Ordering by columns is optional.

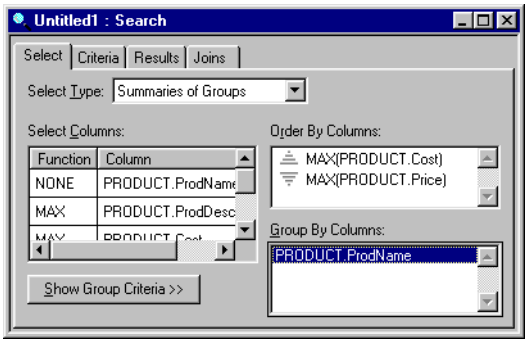
  The order of the columns in the list determines the ordering hierarchy. For example, if the first order column is "state or province" and the second "customer name", the results are first ordered by state or province; customers in the same state or province are then ordered by name.

  The triangle to the left of the column name determines whether the ordering is ascending (▲) or descending (▼). To change the order direction for a column, click the triangle.

### *Summaries of Groups*

The **Summaries of Groups** search type returns summaries for groups of rows with the same values in specified columns. For example, it allows you to find out the total sales for each sales region in an invoices table by selecting the sum of invoice amounts and grouping by sales region.

When you choose **Summaries of Groups**, the following Search action's Select window appears.




- **Select Columns**. Drag into this list the columns you want to select. Select columns for this select type have an associated function. This function is performed on the column for all the rows in a particular group, as determined by the **Group By Columns** list. For example, if you selected the `MAX` function for a "price" column and group by the "classification" column, you would receive one row for each unique classification. Each row would contain the maximum value of the "price" column for the classification being summarized.

The following table lists the available functions.

| Function | Description |
|---|---|
| MAX | The maximum value of column in the group. |
| MIN | The minimum value of column in the group. |
| AVG | The average value of column in the group. Valid only for numeric columns. |
| SUM | The sum of all column values in the group. Valid only for numeric columns |
| COUNT | The number of non-null values in the column for the group. |
| None | Perform no function; return the value of the column for each group. Columns with this option must appear in the **Group By Columns** list, because only group columns are sure to have the same value within a group. |

To choose the function for a column, click the **Function** column and select the function from the drop-down list.

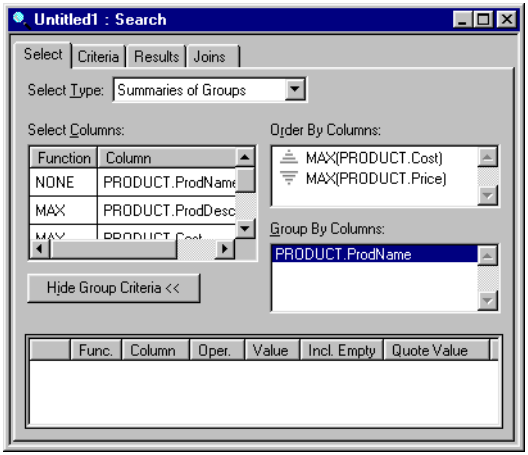
- **Order By Columns**. As with the normal select type, you specify in this list the ordering of results. You can drag columns from the Data Sources Workspace or from the **Select Columns** list. You can order only by columns appearing in the **Select Columns** list.
- **Group By Columns**. The columns in this list determine how rows are grouped before being summarized. Groups consist of all the rows that have the same values in the columns specified. For example, if you group by the “cust_state” and “cust_rep” columns in a customer table, you get one summary row for each group of rows with the same values in the “cust_state” and “cust_rep” columns.
- **Show Group Criteria**. Normally, all the summary rows are returned for records matching the user’s criteria. You can eliminate summary rows by specifying group criteria. The group criteria have a different function from the regular criteria in that the regular criteria specify which rows are eligible for grouping, while the group criteria specify which summary rows are returned.

**Note** The group criteria section is equivalent to the `HAVING` clause in a SQL `SELECT` statement.

For example, if you are grouping by classification and selecting the maximum order amount, you can use group criteria to limit the returned rows to those customers whose maximum order amount is greater than $5,000.

To specify group criteria, choose **Show Group Criteria**.

The Select window expands to show the area for entering group criteria.




Drag columns from either the Data Sources Workspace or the **Select Columns** list.

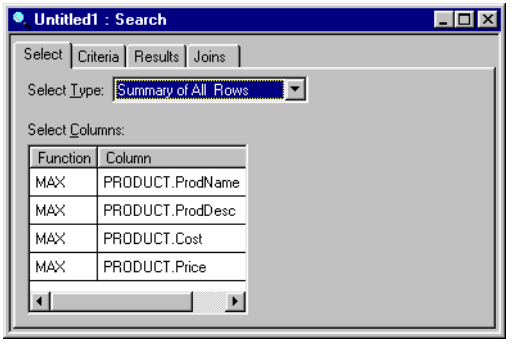
**Note** You can specify *only* columns appearing in the **Select Columns** list.

Except for the function option, you specify group criteria just like normal criteria.

For more information, see “Criteria Tab” on page 234.

### *Summary of All Rows*

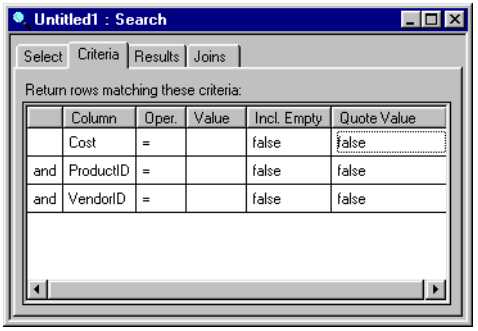
To get a summary of all rows matching a specified criteria, use the **Summary of All Rows** search type. Only one summary row is returned. For example, you could use this search type to find the average amount of all orders in an orders table.




As with the **Summaries of Groups** search type, each select column has an associated function that determines how that column is summarized. All the column values in the rows matching the criteria are aggregated using the specified function.

## Criteria Tab

The Search action criteria determine which rows from the database are returned by the action. If no criteria are specified, all rows are returned; otherwise, each row in the database is compared to your criteria and only those meeting them are returned.



To specify the criteria, drag columns from the Data Sources workspace to the Criteria list. For each column, you need to specify:

- Logical Operator
- Column
- Operator
- Value
- Include Empty
- Quote Value.

### *Column*

In the **Column** field, specify the column whose value you want to compare. Drag the column from the Data Sources Workspace.

### *Logical Operator*

The first field in the criteria list is the logical operator.

You can also right click the field, and choose **Edit** from the context-sensitive menu. Then choose an operator from the drop-down list that appears.

Click the row, then click the field to display a drop-down list to choose an operator from. The operators are **and** and **or**.

**Note** You must specify at least two columns before the logical operators are available.

The logical operator determines whether the current and previous criteria must be true for a record to be included in the result or whether a match on either the current or previous criterion causes a record to be included in the result. For example, if your criteria are:

```
        cust_num      =             5100
and     cust_name     Begins with   A
```

only records matching both criteria are returned. If the logical operator is changed to **or**, records matching either one of the criteria are returned.

There is an implied order of operation for logical operators. Criteria joined with the **and** logical operator are evaluated before those joined with the **or** logical operator. For example, in the following criteria:

```
        cust_num      =             5100
or      cust_name     Begins with   A
and     cust_state    =             NY
```

a match is made if both the second and third criteria are true or the first criterion is true.

For more information about inserting meta tags in entry fields, see "Inserting Meta Tags" on page 113.

You can also use the **Insert Meta Tag** command to enter in the Criteria window entry fields many of the commonly used meta tags.

To insert a meta tag, either click the field and choose **Insert Meta Tag** from the **Edit** menu, or right click the field and choose **Insert Meta Tag** from the context-sensitive menu that appears.

For more information, see "Criteria Separators" on page 239.

You can use criteria separators to control the order of criteria evaluation, regardless of this default logical operator hierarchy.

### *Operator*

In the operator field (**Oper.**), specify the operator to use when comparing the field.

You can also right click the field, and choose **Edit** from the context-sensitive menu. Then choose an operator from the drop-down list that appears.

Click the row, then click the field to display a drop-down list to choose an operator from. Possible operators include:

| Operator | Meaning |
|---|---|
| Is Null | matches an empty field |
| Is Not Null | matches a non-empty field |
| Is In | matches one of a list of values (see page 237) |
| = | is equal to |
| != | is not equal to |
| > | greater than |
| < | less than |
| >= | greater than or equal to |
| <= | less than or equal to |
| Begins with | field begins with these character(s) |
| Ends with | field ends with these character(s) |
| Contains | field contains these character(s) |

Text columns permit the use of any operator; for other columns, the **Contains**, **Begins with**, and **Ends with** operators are disabled.

You can either specify a static operator or insert a meta tag to get the value at execution time. Using a variable operator allows you, for example, to put a drop-down list on your Web page to let users choose the comparison operator.

When using a variable to specify the criterion operator, Tango requires you to use special values to represent each of the operators. The following table lists these special values:

| To Specify This Operator | Use This Value |
|---|---|
| Is Null | inul |
| Is Not Null | nnul |
| Is In | isin |
| = | iseq |
| != | isnt |
| > | gthn |
| < | lthn |
| >= | gteq |
| <= | lteq |
| Begins with | swth |
| Ends with | ewth |
| Contains | cont |

For example, to create an operator drop-down list in an HTML form whose value you want to use as the operator in a search criterion, you could use HTML similar to the following:

```
<SELECT NAME="cust_name_op" SIZE=1>
<OPTION VALUE = "iseq" SELECTED>=
<OPTION VALUE = "isnt">!=
<OPTION VALUE = "gthn">&gt;
<OPTION VALUE = "lthn">&lt;
<OPTION VALUE = "gteq">&gt;=
<OPTION VALUE = "lteq">&lt;=
<OPTION VALUE = "swth">Begins With
<OPTION VALUE = "ewth">Ends With
<OPTION VALUE = "cont">Contains
</SELECT>
```

and set the operator in the Search action to

```
<@ARG cust_name_op>
```

The **Is In** operator needs some additional explanation. It matches records where a column value is in a list of values.

For example, the following criteria:

```
          cust_num      Is in         200, 300, 400
```

matches records in which the `cust_num` field has a value of `200`, `300`, or `400`. The **Is in** operator can be thought of as a shortcut for a series of **OR** equals criteria:

```
          cust_num      =             200
   or     cust_num      =             300
   or     cust_num      =             400
```

The value specified can be a single-column or single-row array (as would be returned by the `<@ARG>` tag with a `type` attribute of `ARRAY`, for example) or a comma-separated list of values.

### *Value*

In the **Value** field, enter the value to use in the comparison.

For more information about inserting meta tags in entry fields, see "Inserting Meta Tags" on page 113.

The value can also contain any value-returning Tango meta tags, which are substituted when the application file is executed. Use the **Insert Meta Tag** command to enter many of the commonly used meta tags.

### *Include Empty*

In the **Incl. Empty** field, specify whether the criterion is included, even if the comparison value is empty.

You can also right click the field, and choose **Edit** from the context-sensitive menu. Then choose a value from the drop-down list that appears.

Click the row, then click the field to display a drop-down list to choose a value from. The values appear as **false** and **true**.

**false** omits the criterion if the value (after meta tag substitution) is empty; **true** includes the criterion regardless of the value's contents.

This option is used mainly for columns whose search value is taken from a search form on a Web page. For example, you may have a search form that allows you to enter search values for several columns, but you want the search done only on the columns you enter values for. To do this, set the **Incl. Empty** option for each of the corresponding Search action criteria to **false**.

There are cases where you do want a criterion included, even if the value is empty. For example, suppose you have a page that asks for a user name and password, and a corresponding Search action that finds the user in a users' database. In the Search action, you probably want to set the **Incl. Empty** option for each of the values to **true**. If you do not, and the user leaves both fields empty, the Search action omits both criteria and returns all user records.

You can right click the **Incl. Empty** field, and choose **Insert Meta Tag** to enter many of the commonly used meta tags.

For more information about inserting meta tags in entry fields, see "Inserting Meta Tags" on page 113.

## Quote Value

In the **Quote Value** field, specify whether Tango puts quotation mark characters around the value in the SQL it generates for this criterion.

Click the row, then click the field to display a drop-down list to choose a value from. The values appear as **false** and **true**.

You can also right click the field, and choose **Edit** from the context-sensitive menu. Then choose a value from the drop-down list that appears.

For text, date, time, and timestamp columns, you should set this option to **true**. For date, time, and timestamp columns, this option has special meaning. **true** converts the specified value from the default Tango format to the format required by the database server; **false** passes the value specified as is to the database server.

If you want to use an expression that the database server evaluates (instead of a literal Tango-supplied value), set the **Quote Value** option to **false** and enter the expression in the **Value** field.

For numeric and Boolean types, you should set the **Quote Value** option to **false**.

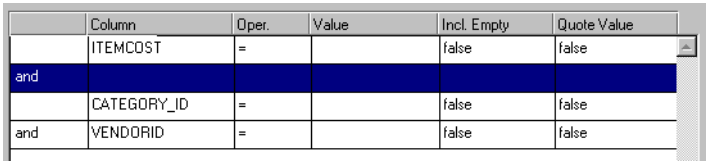
You can right click the **Quote Value** field, and choose **Insert Meta Tag** to enter many of the commonly used meta tags.

For more information about inserting meta tags in entry fields, see "Inserting Meta Tags" on page 113.

### *Criteria Separators*

To group criteria, select the position between the criteria you want to group and choose the **Insert Criteria Separator** command from the **Edit** menu. Only the logical operator cell can be edited for separator items.

You can also right click where you want to insert the criteria separator and choose **Insert Criteria Separator** from the context-sensitive menu that appears.

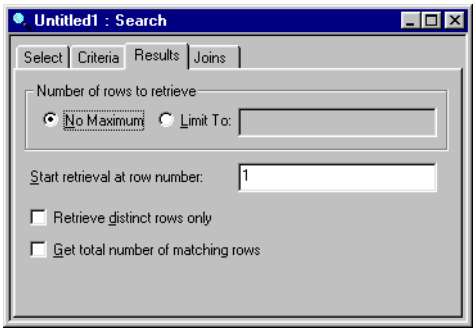


| | Column | Oper. | Value | Incl. Empty | Quote Value |
|---|---|---|---|---|---|
| | ITEMCOST | = | | false | false |
| and | | | | | |
| | CATEGORY_ID | = | | false | false |
| and | VENDORID | = | | false | false |

Upon execution, the criteria before the separator are combined with the criteria after the separator using the logical operator specified in the separator line in the criteria list.

## Results Tab

In the Results window, you specify the maximum number of records to retrieve from the data source, at which result record number retrieval begins, and whether Tango gets the count of matching records.




### *Number of rows to retrieve*

To return all matching records, select **No Maximum**.

To limit how many records you want the search to return, select **Limit To** and enter the maximum number of records to retrieve.

The following options are only available for the **Normal** search type.

### *Start retrieval at row number*

Select this option if you want to skip some of the matching records. Specify the row number you want the Search action to start retrieval at. The default is "1". When the value is other than "1", the search action returns records starting at that number, skipping any records before it.

This option is most useful when you use a variable (such as `<@SEARCHARG start>`) for the starting record number.

For more information, see "Show Multiple Pages If Limit Exceeded" on page 171.

For an example of how to use this option to provide results paging for large result sets, look at the Search action in a Search Builder-generated file created with the **Show Multiple Pages If Limit Exceeded** option selected.

### *Retrieve distinct rows only*

If you select this option, Tango Server adds the `DISTINCT` keyword after the `SELECT` keyword in the generated SQL. The `DISTINCT` keyword specifies whether duplicate rows are to be eliminated from the result set. For example,

```
SELECT c1.cust_state, c1.cust_zip FROM customer c1;
```

becomes

```
SELECT DISTINCT c1.cust_state, c1.cust_zip FROM
customer c1;
```

### *Get total number of matching rows*

You use this option to retrieve the number of records matching the search criteria, irrespective of how many records are actually retrieved.

Using this option, you can access the value in the Search action's Results HTML by using the `<@TOTALROWS>` meta tag.

**Note** Selecting this option involves an extra database operation, so unless you require the information it provides, you should not select it.

## Executing a Search Action

When Tango Server executes a Search action, the search is performed against the associated data source. The result rowset is automatically stored as an array in the local variable `resultSet`. The Results HTML for the Search action is then processed.

For more information, see "<@ROWS> </@ROWS>" on page 121, "<@COL>" on page 58, and "<@COLUMN>" on page 60 of the *Meta Tags and Configuration Variables* manual.

The HTML in the `<@ROWS><@/ROWS>` block, if any, is processed once for each record in the results. Use `<@COLUMN>` or `<@COL>` meta tags to include field values.

If the Search action generates no results, and you have specified No Results HTML for the action, that HTML is processed instead of the Results HTML.

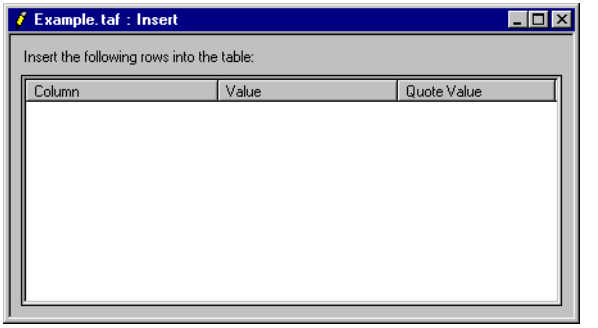
# Adding Records to a Database

The Insert action adds a record to a table in a database.

## Setting Up an Insert Action

When you drag the Insert action icon from the Actions bar into an application file, the Insert action editing window appears.

Insert Action




### *To set up an Insert action*

1. From the Data Sources Workspace, drag into the **Column** list the columns whose values you want to set in the new record.

**Note** You can select columns from only one table.

If you do not add all of the table's columns to the Insert action, the omitted columns are given the default values defined when the database was created.

2. In the **Value** field for each column, enter the value for that column in the new record. The value can contain any of the value-returning Tango meta tags, which are substituted upon execution of the application file.

To insert a meta tag, either click the field and choose **Insert Meta Tag** from the **Edit** menu, or right click the field and choose **Insert Meta Tag** from the context-sensitve menu that appears.

For more information about inserting meta tags in entry fields, see "Inserting Meta Tags" on page 113.

The **Quote Value** option operates the same as it does in search criteria.

For more information, see "Criteria Tab" on page 234.

## Executing an Insert Action

When Tango Server executes an Insert action, a record is added to the database with the column values specified. The Insert action returns no results.

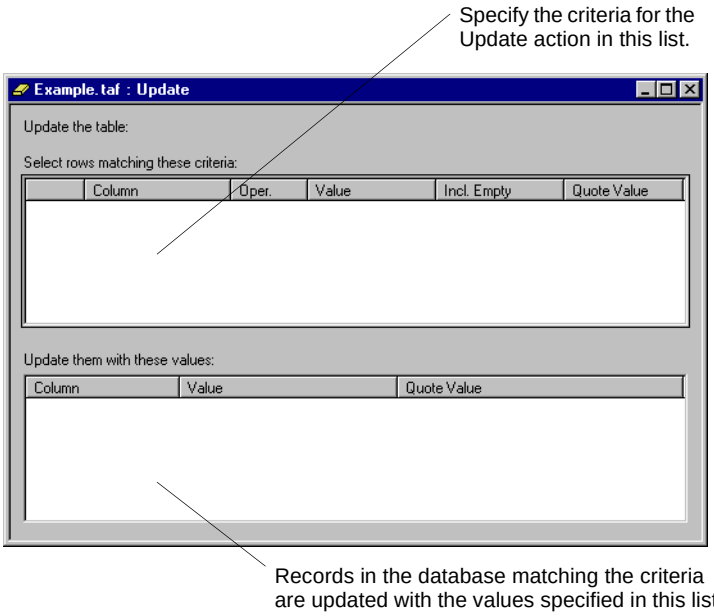
# Modifying a Database Record

The Update action modifies database records matching specified criteria.

## Setting Up an Update Action

When you drag the Update action icon from the Actions bar into an application file, the Update action editing window appears.

Update Action




### *To set up an Update action*

**1** In the criteria list at the top of the action's editing window, specify which records you want to update.

You edit the criteria list the same way you edit the Search action's criteria list.

For more information, see "Criteria Tab" on page 234.

**Caution** For an Update action, be *extremely* careful when setting the **Incl. Empty** option to **false**. You may end up affecting more rows than you intend, possibly even updating all the records from your database table. Just like leaving **Incl. Empty** set to false in a Search action returns all the records, leaving it to false in an Update action updates all records.

2 From the Data Sources workspace, drag the columns whose values you want to update into the update columns list at the bottom of the action's editing window.

**Note** You can specify columns from only one table. If you want to update multiple tables, use an Update action for each table. In this case, consider using a Transaction action to make sure all or none of the updates are processed.

Only the values in the columns you specify are modified when the action is executed.

3 Under **Value** for each column, enter the new value for that column.

The value can contain any of the value-returning Tango meta tags, which are substituted upon execution of the application file.

To insert a meta tag, either click the field and choose **Insert Meta Tag** from the **Edit** menu, or right click the field and choose **Insert Meta Tag** from the context-sensitve menu that appears.

For more information about inserting meta tags in entry fields, see "Inserting Meta Tags" on page 113.

If you always want to update a column with a fixed value, simply enter that value.

The **Quote Value** option operates in the same way it does in search criteria.

For more information, see "Quote Value" on page 239.

## Executing an Update Action

When Tango Server executes an Update action, Tango searches for records matching the specified criteria and updates them with the specified column values.

The Update action returns no results.

# Removing a Database Record

The Delete action removes database records that match the specified criteria.

For more information, see “Criteria Tab” on page 234.

You edit the criteria list the same way you edit the Search action’s criteria list.

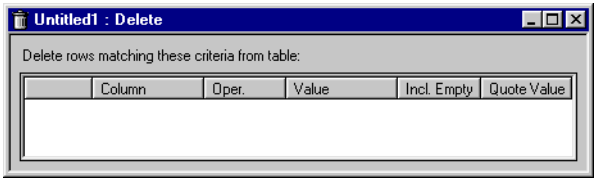
## Setting Up a Delete Action

Delete Action

### *To set up a Delete action*

1. Drag the Delete action icon from the Actions bar into an application file.

   The Delete action editing window appears.




2. In the criteria list of the Delete action’s editing window, specify which records you want to delete.

**Note** You can specify columns from only one table. If you want to delete multiple tables, use a Delete action for each table. In this case, consider using a Transaction action to make sure all or none of the deletes are processed.

You must specify at least one criterion for the Delete action to be valid.

**Caution** For a Delete action, be *extremely* careful when setting the **Incl. Empty** option to **false**. You may end up affecting more rows than you intend, possibly even deleting all the records from your database table. Just like leaving **Incl. Empty** set to false in a Search action returns all the records, leaving it to false in a Delete action deletes all records.

## Executing a Delete Action

When Tango Server executes a Delete action, records matching the specified criteria are deleted.

The Delete action returns no results.

# Adding Custom Columns to Database Actions

A custom column entry lets you enter any text as the column reference. You can use custom columns wherever Tango accepts columns dragged from the Data Sources Workspace.

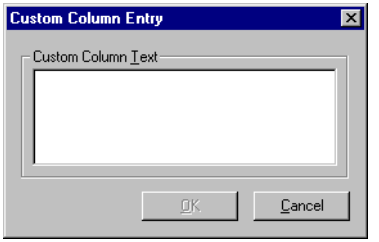
Make sure the text entered makes sense in the database action. For example, in a Search action, you could enter the following calculation as a Select column:

```
orders.order_amt + 20
```

### *To add a custom column to a database action*

1. Do either of the following:
   - From the **Edit** menu, choose **Insert Custom Column**.
   - Right click any database action editing window where you can add columns and choose **Insert Custom Column** from the context-sensitive menu that appears.

   The Custom Column Entry dialog box appears.




2. Enter the text to use as the column reference.
3. Choose **OK**.

Custom columns can be edited later by double clicking the column reference in the list.

CHAPTER FOURTEEN

# Using Control Actions

14

*Setup and Operation of Branch, If, Loop, Break, and Return Actions*

Tango application files can contain control actions that control the execution of other actions in the file.

This chapter covers the setup and operation of the following actions.

- The *Branch* action causes a jump to a designated action or group.
- The *If* action evaluates an expression and based on the result of that expression affects the control flow of the file.
- The *Loop* action repeats a set of contained actions a given number of times or while an expression evaluates to true.
- The *Break* action terminates processing in the loop.
- The *Return* action ends execution of the application file and returns any accumulated Results HTML to the Web browser. It can also return to another application file.

# Jumping to a Designated Action (Branch Action)

The Branch action causes a jump to a designated action or to an action group.

You can set the Branch action to jump to a different action in the same file the Branch action is in or an action in another file. There are restrictions on what position and kinds of action a Branch action can jump to.

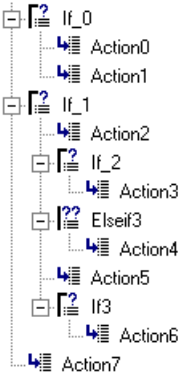
## Branch Action Destination Rules

Branch actions have the following rules for valid destinations:

- Branch actions can only branch to an action at the outermost level in a different file.
- Branch actions are allowed to branch to any action at the same level in the same block of Tango actions.
- Branch actions are allowed to branch to any action at the outermost level.
- Branch actions are allowed to branch to any action that is an immediate parent.
- Branch actions are allowed to branch to an action that is a first-level child of an immediate parent.
- Branch actions cannot branch to Else If or Else actions.

The preceding rules mean that Branch actions cannot branch to an action that they dominate or that a sister action dominates.

For example, look at this application file:



Examples of valid branches are:

- Action2 can branch to Action5.

  The actions are in the same action block and at the same level.

- Action2 can branch to Action7.

  Action7 is at the outermost level.

- Action3 can branch to If_2 and If_1.

  Those actions are immediate parents of Action3.

- Action3 can branch to Action2.

  Action2 is the first-level child of an immediate parent of Action3, If_1.

Examples of invalid branches are:

- Action1 cannot branch to Action4.

  They are not in the correct relationship: Action4 is downward from Action1.

- Action2 cannot branch to Action6.

  They are not in the correct relationship: Action6 is downward from Action2.

- Action2 cannot branch to ElseIf3.

  Even though ElseIf3 is the first child of a immediately dominating parent action (If_1), this branch is not allowed because branching to an Else If action is invalid.

## Branching to Other Application Files When Caching is On

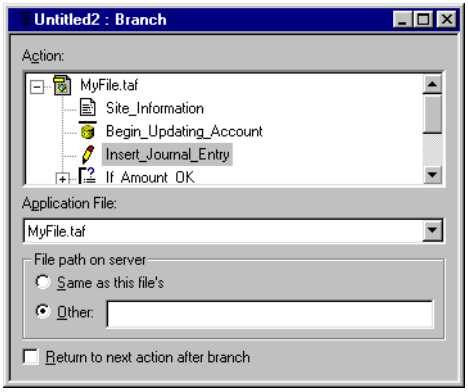
If a Branch action is used to branch to another application file when document caching is on (as it is by default), changes to the branched-to application file are ignored; that is, the cached version of the file is executed when the Branch action executes, rather than the new one. This may affect ease of development of the application files involved; it may be helpful to do one of the following:

- Disable document caching while developing application files that branch to other files. Open the configuration application file with your browser, click **Memory**, and disable the **Cache Files** option.
- If you modify a branched-to application file, also make a change to the document containing the Branch action and save it. Both application files will be reloaded in this case.

## Setting Up a Branch Action

Branch Action

When you drag the Branch action icon from the Actions bar into an application file, the Branch action editing window appears.




### *To set up a Branch action*

1. From the **Application File** drop-down list, select the file you want the Branch action to jump to.

   If the selection is **Current**, you select the target action from the application file you added the Branch action to.

   If the current file is part of a project, the list also shows the other files in the project.

2. If you want to select an application file from your hard drive, choose **Others** from the drop-down list.

   A standard file selection dialog box appears to select an application file with.

   If you select an application file that is not the current file, the Action list changes to show the actions in the selected file. The **File path on server** section is also enabled to allow you to specify the path to the application file.

3. Do either of the following:

   - Select the **Same as this file's** option (the default) to cause Tango Server to always look in the current file's folder.
   - Specify in the **Other** field the path to the desired folder, which causes Tango Server to look for the application file in that location. This path is specified relative to the Web Server's document root directory.

4. Enable the **Return to next action after branch** option if you want execution to continue with the action following the Branch action when a Return action is encountered in the destination. Disable this option if you want execution to end when a Return action is encountered in the destination.
5. In the **Action** list, select the action you want the branch action to jump to.

The destination action for a Branch must be valid according to the rules on page 248; otherwise, an error is returned.

## Executing a Branch Action

When Tango Server executes a Branch action, it jumps to the designated action. If the **Return to next action after branch** option is selected, Tango returns to the action following the Branch action when a Return action is encountered.

# Conditional Action Execution (If Action)

The If action evaluates an expression and based on the result of that expression affects the control flow of the file. The If action also has two related actions called Else If and Else.

The If action can exist with or without one or more Else If actions and with or without a single Else action. The general forms of these actions are as follows:

- **If Action**

  If action (*expression*)

  group of actions to execute
  if *expression* evaluates to *true*

  remainder of actions in file

- **If and Else Actions**

  If action (*expression*)

  group of actions to execute
  if *expression* evaluates to *true*

  Else action

  only evaluated if If action evaluates to false

  group of actions to execute
  if *expression* evaluates to *false*

  remainder of actions in file

- **If, Else If, and Else Actions**

  If action (*expression*)

  group of actions to execute
  if *expression* evaluates to *true*

  Else If action (*expression*)

  only evaluated if If action evaluates to false

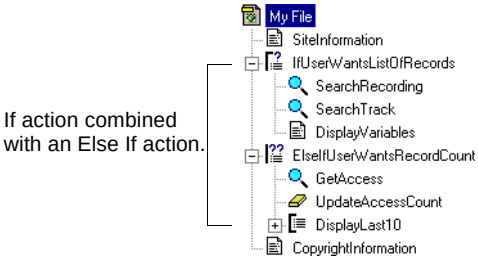
  group of actions to execute
  if Else If *expression* evaluates to *true*

  Else action

  group of actions to execute
  if none of the If or Else If *expression*s
  evaluate to *true*

  remainder of actions in file

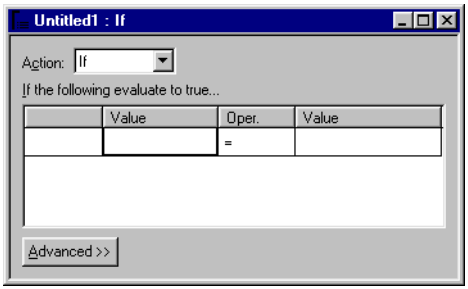
The following is an example of how If and Else If actions appear in an application file.




## Setting Up an If Action

If, Else If, and Else Actions

When you drag the If or Else If action icon from the Actions bar into an application file, the If action editing window appears.




By default, the If action editing window appears in its *basic* view, allowing you to create expressions quickly. An expression line appears in the dialog box, ready for you to edit.

**Note** Dragging the Else action icon into an application file does not open any action editing window. However, if you double click an Else action in an application file, the action editing window opens so you can change the Else action to an If or Else If action.

For more information, see “Advanced View” on page 255.

An *advanced* view is also available that gives you more flexibility than the basic view when specifying evaluation expressions.

You change the type of If action by selecting **If**, **Else If**, or **Else** from the **Action** drop-down list.

The If and Else If action editing windows are basically the same, and you enter evaluation expressions the same way for both of them. When you select **Else**, however, only the **Action** drop-down list is active. This allows you to change to another If action type.

**Note** If you change back to an If or Else If action type from an Else action type before closing the action editing window, any If or Else If expressions you specified previously are retained.

## Basic View

### *To specify values for the basic view parameters*

Specify values as follows:

- **Logical Operator**. The first field in the expression list is the logical operator. There are two logical operators: **and** and **or**.

  Click the row, then click the field to display a drop-down list to choose an operator from. The logical operator is used when the expression includes more than one row.

For more information, see “Logical Operator” on page 235.

- **Value**. Enter the values to use in the expression. The values can contain any value-returning Tango meta tags, which are substituted when Tango Server executes the action.
- **Oper**. Specify the operator to use to compare the two values specified.

For more information, see “Operator” on page 236.

  You can also use the **Insert Meta Tag** command to enter many of the commonly used meta tags.

For more information about inserting meta tags in entry fields, see “Inserting Meta Tags” on page 113.

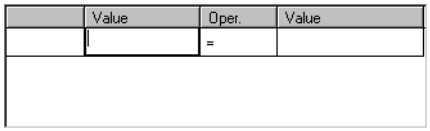
To insert a meta tag, either click the field and choose **Insert Meta Tag** from the **Edit** menu, or right click the field and choose **Insert Meta Tag** from the context-sensitive menu that appears.

### *To add a new parameter row*

Do one of the following:

- From the **Edit** menu, choose **Insert**.
- On the main toolbar, click the Insert icon.
- Right click the list area, and choose **Insert** from the context-sensitive menu that appears.

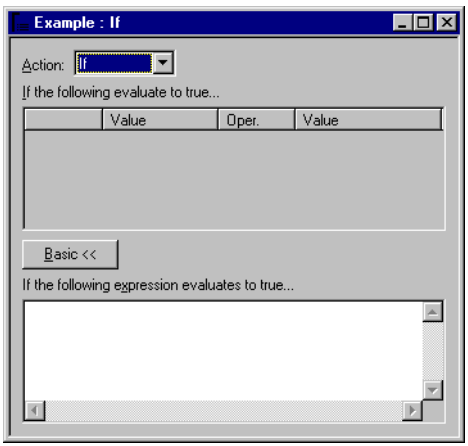
A blank row appears.




## Advanced View

When you click **Advanced** in the basic view, the following happens:

- The window expands to show a text area where you can enter text-based expressions.
- The expression list in the basic view is disabled (appears grayed).
- The **Advanced** button changes to **Basic**.
- Any expression in the basic list is automatically converted to a text expression in the advanced text area.




The advanced view presents a free-form text area to give you more flexibility than the basic view when specifying evaluation expressions. For example, if you want to use parentheses to control the evaluation order, you can enter the expression in this area.

For more information, see "<@CALC>" on page 38 of the *Meta Tags and Configuration Variables* manual.

The expression entered here takes the same form as expressions specified for the `<@CALC>` meta tag.

If the expression evaluates to "1" or "true", the expression is considered to be true; otherwise, it is considered to be false.

You can automatically regenerate an expression appearing in the basic view in the advanced view as follows:

1. In the advanced text area, right click where you want the expression to appear, or select the text you want to replace and right click.
2. From the context-sensitive menu that appears, choose **Insert Expression As Above**.

   The expression appears in text form in the advanced text area.

**Tip** You can also drag text from the Snippets Workspace to this text area.

To return to the basic view, click **Basic**.

**Caution** If you changed the expression in the advanced view, your changes are lost when you return to the basic view. An alert box asks if you want to continue.

## Performing Operations on If Actions

Working with If actions is similar to working with grouped actions.

For information on the operations you can perform on groups, see “Working With Action Groups” on page 223.

## Executing an If Action

If Tango Server evaluates the expression to true, action processing continues with the list of indented actions.

If the expression evaluates to false, action processing jumps to the first action after the list of indented actions, which could be an Else or Else If action.

If that action is an Else If action, Tango Server evaluates the Else If expression to determine whether it should execute the indented actions under the Else If action.

If the action is an Else action, then Tango Server executes the actions indented under the Else action.

Once processing of the indented actions is complete, processing continues with the next non Else If or Else action at the same level as the If action.

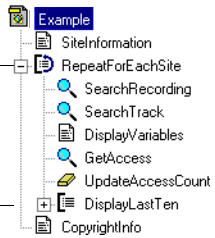
**Tip** If actions may be nested; that is, the actions nested under an If action may contain other If actions.

# Repeating Actions (Loop Actions)

Loop actions repeat the execution of a set of contained actions for a given number of times or while an expression evaluates to true.

The following is an example of how a Loop action appears in an application file.

Loop action containing a series of other actions.




**Tip** The action list can contain nested Loop actions, that is, loops within loops.

For more information, see “Exiting a Loop (Break Action)” on page 261.

Tango also includes a Break action you can use to exit a Loop action before the loop conditions for termination are met.

For more information, see “For Loop” on page 259.

There are two kinds of Loop actions: While Loops and For Loops.

## Setting Up Loop Actions

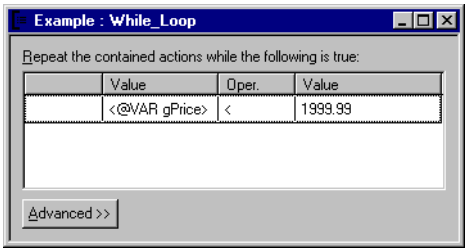
While Loop Action

### While Loop

A While Loop action executes the actions in the loop *while* an expression evaluates to true.

When you drag the While Loop action icon from the Actions bar into an application file, the While Loop action editing window

appears in its *basic* view, allowing you to create evaluation expressions quickly.




If the expression you specify does not evaluate to true when Tango first executes the While Loop action, the enclosed actions are never executed.

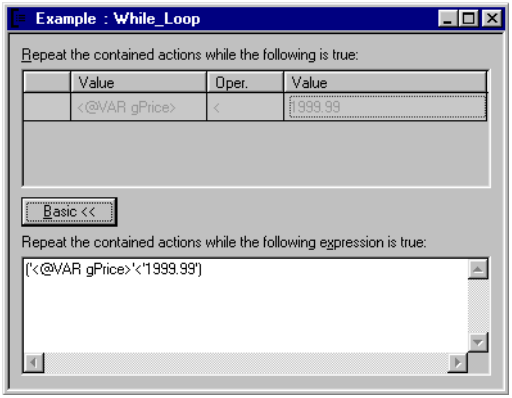
**Caution** Make sure that at least one value being compared in the expression is being changed inside the loop. If this is not the case, a "true" evaluation causes Tango to execute the enclosed actions forever.

Be careful when constructing a While Loop expression. You want to ensure that it eventually evaluates to false.

For more information, see "Basic View" on page 254.

The basic view for a While Loop action is similar to the basic view for If and Else If actions.

The While Loop action editing window also has an advanced view that gives you more flexibility than the basic view when constructing evaluation expressions. For example, you can use parentheses to control the evaluation order.




For more information, see "Advanced View" on page 255.

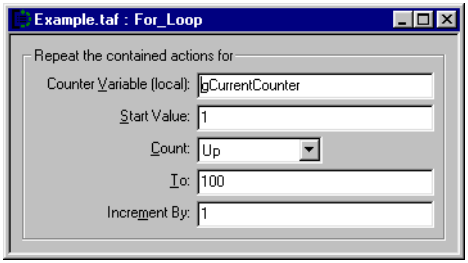
This view is similar to the advanced view for If and Else If actions.

## For Loop

The other loop action is the For Loop action, which repeats a set of contained actions *for* a given number of times.

For Loop Action

When you drag the For Loop action icon from the Actions bar into an application file, the For Loop action editing window appears.




Set the parameters in a For Loop action as follows:

- **Counter Variable (local)**. The name of a local variable to use to access the current value of the counter.

**Note** This parameter is optional. It is not required to use the action.

- **Start Value**. The starting value for the loop counter. The default is 1.
- **Count**. The direction of the counting from the starting value to the ending value. You must specify this parameter so Tango can increment or decrement the counter properly. Choose **Up** or **Down** from the drop-down list to set the counter to increment or decrement, respectively. The default is **Up**. When **Down** is selected, the **Increment By** field name becomes **Decrement By**.
- **To**. The ending value for the loop counter.
- **Increment/Decrement By**. The value the counter increments or decrements by on each loop.

**Tip** All For Loop action fields, except for the **Count** field, support Tango meta tags.

## Performing Operations on Loop Actions

Working with Loop actions is similar to working with grouped actions.

For information on the operations you can perform on grouped actions, see “Working With Action Groups” on page 223.

## Executing Loop Actions

### While Loop

Tango Server evaluates the expression before executing the contained actions. If the expression evaluates to true then the contained actions are executed. Then, Tango returns to the While Loop action and re-evaluates the expression. If it is true, the contained actions are executed again. This process continues until the expression evaluates to false. Execution then continues at the next action outside the loop. If Tango Server finds the expression is invalid, it returns a runtime error.

Tango Server evaluates any meta tags in the expression on each pass through the loop.

### For Loop

If **Start Value** is a meta tag, Tango Server evaluates it prior to the first pass through the loop.

If the **To** and **Increment/Decrement By** fields contain meta tags, Tango Server evaluates them on each pass through the loop.

At the end of the loop, execution returns to the top. This process continues until the loop counter exceeds the value specified in the **To** field. Execution then continues with the next action outside the loop.

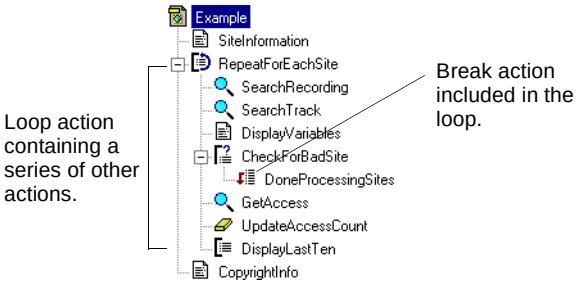
# Exiting a Loop (Break Action)

The Break action prematurely terminates processing in a loop or group action. On termination, processing continues at the next action after the loop or group.

Break Action

Drag the Break action icon from the Actions bar into a Loop or Group action at the point you want the loop or group to terminate.

The following is an example of an application file showing how Loop and Break actions appear.




On execution, the Break action terminates the loop, and processing continues at the next action after the loop.

**Note** If you include a Break action outside a loop or group, on execution Tango Server generates a runtime error.

# Ending File Processing (Return Action)

Return Action

The Return action ends application file processing and returns any accumulated Results HTML to the Web browser.

For more information, see “Jumping to a Designated Action (Branch Action)” on page 248.

The exception to this is if the current execution flow is the result of a Branch that had its **Return to next action after branch** option set. In this case, the execution returns to the action following the Branch when a Return action is encountered.

CHAPTER FIFTEEN

# Extending Tango Functionality

15

*Script and External Actions*

The *Script* action provides an interface within Tango for executing JavaScript code at the server. The script executed can return to Tango a value you can access using Results HTML.

The *External* action calls an executable invoked using a command line, a Dynamic Link Library (DLL), or a Java class file to perform processing and, if desired, return results. Under Mac OS, the External action can also send a specified Apple Event with parameters to a specified application and retrieve the results, or use Apple Event actions to communicate with applications you write in AppleScript.

This chapter covers the following topics:

- setting up and executing a Script action
- configuring a DLL call or a Java action
- using a command line
- assigning variables to action parameters
- assigning action attributes
- deleting action parameters
- executing an External action.

# Executing JavaScript

The Script action provides an interface for executing core JavaScript code at the server. The script executed can return a value to Tango, which is accessible using Results HTML.

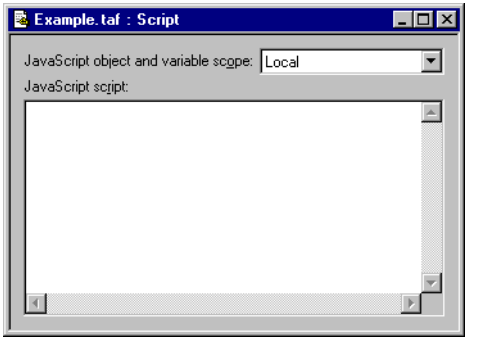
Tango is JavaScript 1.2 compatible, meaning it includes the official JavaScript Reference implementation from Netscape, and conforms to version 1.2 of the language.

**Note** Tango supports the general purpose core of the language. The objects representing the Web browser and its contents are *not* supported, because the scripts are executed at the server where these objects do not exist. For your convenience, a JavaScript HTML reference is provided with the Tango HTML help.

## Setting Up a Script Action

Script Action

When you drag the Script action icon from the Actions bar into an application file, the Script action editing window appears.




The **JavaScript object and variable scope** parameter defines the lifetime of the objects and functions declared in the script. This is a concept similar to the scope of variables in Tango. The drop-down menu has two choices:

- **Local** (the default) specifies that anything defined in the script can be referenced in another script in the same application file execution.
- **Immediate** specifies that the objects and functions go away immediately after the action is executed.

For information on using the script text area, see "HTML Editing Window" on page 24, and "The SQL Query Window" on page 35.

You enter the text script to be executed in the **JavaScript script** text area. The script may contain meta tags. All meta tags in the script are substituted before the script is executed.

The script text area functions the same as HTML text editing windows.

When you right click the script text area, the same context sensitive menu for the SQL text area of a Direct DBMS action appears.

## Executing a Script Action

For more information, see "<@SCRIPT>" on page 124 of the *Meta Tags and Configuration Variables* manual.

Tango Server executes the Script action in a similar way it executes the `<@SCRIPT>` meta tag.

Any value returned by a script is accessible in the Results HTML by using `<@COL 1>` inside a `<@ROWS>` block. The action's result set (a `1 x 1` array) is also stored automatically in a local variable, `resultSet`.

# Using an External Action

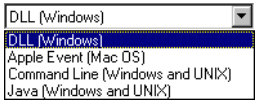
## Setting Up an External Action

When you drag the External action icon from the Actions bar into a file, the External action editing window automatically opens.

External Action

The standard External action type is based on the current platform, in this case, **DLL**. You can also specify Java, command line execution or Apple Event.

From the **Type** drop-down list, select the type of action you want to execute.




**Note** If you specify parameters for one type and then change to another type, Tango attempts to transfer the current parameters to the new type.

See the *Tango Enterprise User's Guide (Mac OS)* for a description of the Apple Event External action.

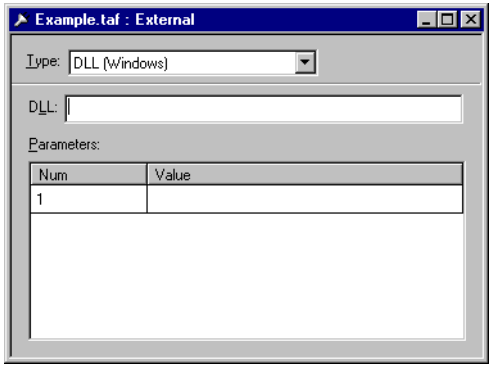
This *User's Guide* gives a description of each type of External action. The **DLL**, **Command Line** and **Java** externals are available if you plan to deploy your application file on Windows. **Command Line** and **Java** are available for Unix, and an **Apple Event** action can only be used with the Mac OS version of Tango Server.

## Configuring a DLL Call

### *To configure a DLL call*

1 From the **Type** drop-down list, select **DLL**, if it is not already selected.

The External action editing window for the DLL type appears.



2 In the **DLL** field, type the fully qualified path to the DLL you want to call, for example,

`C:\Program Files\Tango3\externals\Test.dll`.

This path may contain meta tags.

**Note** The DLL called by the External action must be written to conform to the API described in Appendix B.

3 Insert a new parameter row by doing one of the following:

- From the **Edit** menu, choose **Insert**.
- On the main toolbar, click the Insert icon.
- Right click the parameters area, and choose **Insert** from the context-sensitive menu that appears.

A new parameter row appears. The parameters are numbered for easy identification.

**Tip** You may re-order parameters by dragging them within the list.

4 In the **Value** field, type a parameter value.

Parameter values may include any value-returning Tango meta tags, which are substituted when the action is executed.

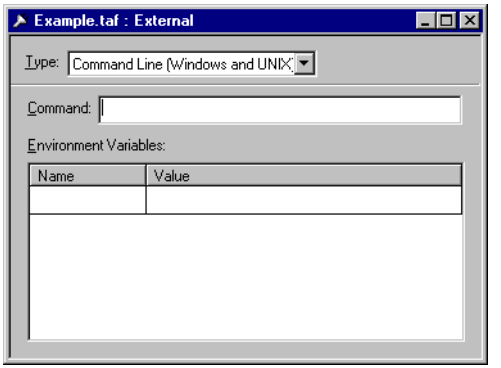
## Using a Command Line

External actions allow you to run any executable file (for example, batch file, shell or Perl script, C application) and, if desired, retrieve results. Values are passed from Tango to the executable by means of environment variables. Results are retrieved by Tango from text that the external action has written to the standard output stream (stdout). Rows and columns are delimited with tabs and returns, respectively.

### *To configure a command line External action*

1 From the **Type** drop-down list, select **Command Line**.

The External action editing window for the Command Line type appears.




2 In the **Command** field, specify the executable file name.

With Windows versions, the value of this field (after meta tag substitutions) must be a valid file path (for example, `c:\temp\dir.bat`). Command line parameters are not allowed here.

You may include command line switches here (Unix only).

**Note** A command line containing a DOS-like command (for example, `dir`) will not work because it is not a file. Commands calling other command processors (shell) also will not work. If you want to execute an operating system command, you should create a batch file or shell script.

On Unix, shell scripts must have read / execute permission for the user running the Tango daemon. In order to be properly executed under the appropriate shell, the shell script must have a shell execution directive such as `#!/bin/sh` as its first line..

3 Insert a new environment variable row by doing one of the following:

- From the **Edit** menu, choose **Insert**.
- On the main toolbar, click the Insert icon.
- Right click the environment variables area, and choose **Insert** from the context-sensitive menu that appears.

A new environment variable row appears.

4 In the **Name** field, enter the name of an environment variable to create for the process. This value is passed on the command line to the External action.

**Note** The name of an environment variable is case sensitive.

5 In the **Value** field, enter the value to assign to the named environment variable.

Environment variables may include any value-returning Tango meta tags, which are substituted when the action is executed.

For more information, see "Assigning Attributes" on page 272.

**Tip** You may re-order variable rows by dragging them within the list.

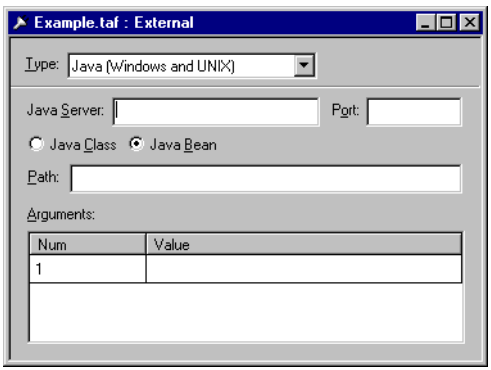
## Configuring a Java Action

Tango ships with its own Java server. The Java server is on the same machine as Tango Server. The Java type external action allows you to connect to the Java server.

### *To configure a Java external action*

1 From the **Type** drop-down list, select **Java**.

The External action editing window for the Java type appears.




2 In the **Java Server** field, type the IP address or *hostname.domain* of the machine running the Java server.

The default server name is **localhost** (127.0.0.1), meaning that the Java server is on the same computer as Tango.

3 In the **Port** field, type the port number the Java server is listening on.

4 Select either **Java Class** or **Java Bean** to specify the type of Java file.

5 In the **Path** field, type the fully qualified path to the Java file. This path can contain meta tags.

6 Insert a new argument row by doing one of the following:

- From the **Edit** menu, choose **Insert**.
- On the main toolbar, click the Insert icon.
- Right click the arguments area, and choose **Insert** from the context-sensitive menu that appears.

A new argument row appears. The arguments are numbered for easy identification.

**Tip** You may re-order arguments by dragging them within the list.

7 In the **Value** field, type an argument value.

Appendix C provides additional information on calling Java classes from Tango.

Arguments may include any value-returning Tango meta tags, which are substituted when the action is executed.

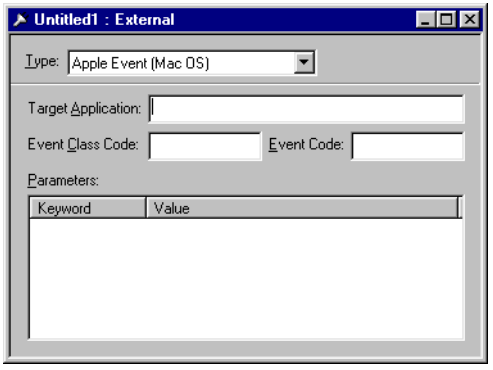
## Configuring an Apple Event (Mac OS only)

### *To configure an Apple Event external action*

To use External actions you should be familiar with Apple Events and know the event and parameter codes of applications with which you want to communicate. You can also use Apple Event actions to communicate with applications you write in AppleScript.

From the **Type** drop-down list, select **Apple Event**.

The External action editing window for the Apple Event type appears.



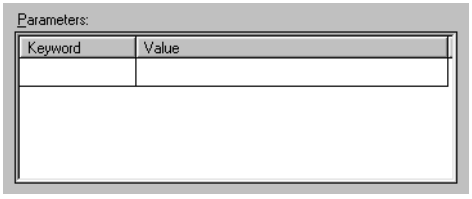
The Apple Event dialog box has the following components:

- **Target Application**. The path to the application to send the Apple Event to. The path must be colon (:) separated, for example: `Mac HD:My Folder:My Script`.
- **Event Class Code.** The case-sensitive, four-letter class code of the event to send to the target application.
- **Event Code.** The case-sensitive, four-letter code of the event to send to the target application.
- **Event Parameters**. Parameter keyword codes are all case-sensitive, four-letter values defined by the target application. To specify the direct parameter for an event, use four dashes, "----".

To add a parameter to the list, do either of the following:

- From the **Edit** menu, choose **Insert**.
- Place the cursor inside the **Parameters** area, right click, and choose **Insert** from the context-sensitive menu.

A blank row appears in the Parameters area.




The parameter values can be entered as fixed values, or you can use any Tango meta tags that return values.

Tango can send only text parameters and accepts only a string (or value that can be converted to a string), list of strings, or list of lists of strings as return values.

If you need to communicate with applications requiring non-string parameters or that return results in a format Tango cannot handle, you should use an intermediate AppleScript application.

### *Calling AppleScript Applications*

For more information on Apple Events and AppleScripts, consult your Macintosh documentation.

AppleScript application handlers that are to be called from Tango require special subroutine names and parameter labels. Here is an example:

```
on «event TNGOTest» dir_param given «class arg1»:arg_1,
«class arg2»:arg_2

set col_one to¬
"Here is argument 1, accessible with a <@COL 1> tag:
" & arg_1
set col_two to¬
"Here is argument 2, accessible with a <@COL 2> tag:
" & arg_2
set col_three to¬
with a <@COL 3> tag: "¬& dir_param

return {col_one, col_two, col_three}

end «event TNGOTest»
```

The first four letters of the AppleScript handler name (after the event keyword) make up the **Event Class Code**; the second four are the **Event Code**. In this example, the event class is `TNGO` and the event is `Test`.

Except for the direct parameter, which immediately follows the handler name, the parameters must be labeled with the class keyword and a unique four letter code. In this example there are three parameters: the direct parameter, “arg1”, and “arg2”.

As with any other applications Tango communicates with through Apple Events, results from an AppleScript application must be returned as a string, list of strings, or list of lists of strings. Here is an example of each:

- `"value"`
- `{"value1", "value2", "value3"}`
- `{{"row1value1", "row1value2", "row1value3"}, {"row2value1", "row2value2", "row2value3"}}`

## Assigning Attributes

You can also assign the following attributes to an External action:

- Results HTML
- No Results HTML
- Error HTML.

You assign these attributes using the **Attributes** menu.

You can also right click the External action icon or name in the application file, or click the open External action editing window to display a context-sensitive menu of available attributes. The **Properties** command is also available in this menu.

For a description of each attribute, see “Assigning Attributes to Actions” on page 214.

If you right click an action parameter, the **Edit** command is active allowing you to edit the parameter value; otherwise, it is disabled (grayed).

## Deleting Parameters

You can also right click the parameter, and choose **Delete** from the context-sensitive menu that appears.


Delete

***To delete an External action parameter***

1. Open the External action editing window.
2. Select the parameter row you want to delete, and do one of the following:
   - Click the Delete icon on the main toolbar.
   - From the **Edit** menu, choose **Delete**.
   - Press the DELETE key.
3. When you are asked to confirm deletion of the selected row, choose **OK**.

## Executing an External Action

When Tango Server executes an External action, the DLL, command line, Apple Event, or Java specified is called and the parameters specified are passed to it. Any data returned is accessible with `<@COL>` tags in a `<@ROWS></@ROWS>` block in the action's Results HTML. For example, if a row set with three columns is returned, all the results can be returned with the following Results HTML:

```
<@ROWS>
Row<@CURRROW>,Column1:<@COL 1><BR>
Row<@CURRROW>,Column2:<@COL 2><BR>
Row<@CURRROW>,Column3:<@COL 3><BR>
<HR>
</@ROWS>
```

For DLLs, rows and columns are determined by the interface defined in Appendix B. For the command line and Java options, the value returned is treated as tab and return delimited: a tab separates columns and a return separates rows.

Only textual data can be returned from an External action.

The entire result rowset from an External action is automatically assigned to a local array variable, `resultSet`.

If the External action generates no data, and you have specified No Results HTML for the action, that HTML is processed instead of the Results HTML.

## Disabling JavaScript, Java and External Actions

For more information, see “Configuring Tango Server” on page 317.

JavaScript, Java, and External actions are by default enabled in Tango. If you want to disable (or enable) these features, you can do so by changing the following options in the `config.taf` application file, in the **Feature Switches** screen:

```
javaScriptSwitch
javaSwitch
externalSwitch
```

For more information, see “Feature Switches” on page 321.

Check or uncheck the checkboxes beside the options.

CHAPTER SIXTEEN

# Sending Electronic Mail From Tango

16

## Mail Action

The *Mail* action sends out electronic mail using the Simple Mail Transport Protocol (SMTP). SMTP is the main protocol used to send mail on the Internet.

The Mail action lets you build features into Tango application files so you can send e-mail messages. For example, you might send a mail message to a list of recipients notifying them of a change to a database or that a particular file has been executed. Many types of information gathering are possible. For example, you can use mail for inventory management, shipping and receiving, data compilation, generating sales leads, or any function that can use data derived from activity in a database.

In fact, you can send a mail message about any subject to anyone with an e-mail address. You simply add the Mail action to an application file, prepare your message, and execute the file to send the message.

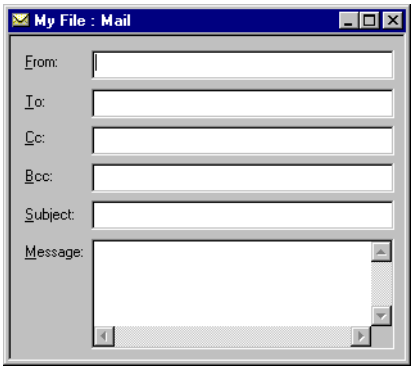
This chapter covers the following topics:

- setting up a Mail action
- getting additional functionality from Tango mail.

## Setting Up a Mail Action

Mail Action

When you drag the Mail action into your application file, the Mail action editing window appears.




Enter your information in the six mail fields as follows:

- **From**. Type the Internet mail address of the person who the mail message is from. Normally, this is also the address that replies and error messages go to.

  If this field is left empty, Tango uses the system configuration variable `mailDefaultFrom` to determine the default value.

  The value of this variable can be changed in the `config.taf` application file. You would enter the address of the person sending the mail message, for example, `tango@example.com`. This variable is also stored in the Tango Server configuration file (`t3server.ini`) as `MAILDEFAULTFROM`.

For more information, see “mailDefaultFrom” on page 213 of the *Meta Tags and Configuration Variables* manual.

For more information, see “Configuring Tango Server” on page 317.

**Note** If both the configuration variable and the **From** field are empty, then the mail message cannot be sent. A mail message must always be *from* somebody.

- **To**. Type an Internet mail address or a comma-delimited list of addresses.
- **Cc** (carbon copy). Type the Internet address of the person you want to send a copy of the message to. This field also allows you to enter a comma-delimited list of addresses.

- **Bcc** (blind carbon copy). This field is the same as **Cc** except the recipients are not listed in the message; that is, the **To**, **Cc** and other **Bcc** recipients will not know that the message also went to those addresses.
- **Subject**. Type the subject of the mail message.
- **Message**. Type your message.

All of the mail fields support the use of Tango meta tags, which are evaluated at the time mail is sent.

When the Mail action is executed, Tango Server connects to the SMTP server. Tango then sends the message to all the specified recipients.

The SMTP server is defined by the configuration variables `mailServer` and `mailPort`. These variables can be changed by using the config.taf application file. They are also stored in the Tango Server configuration file (`t3server.ini`) as `MAILSERVER` and `MAILPORT`.

The result of the action is a one-column array of the messages received from the SMTP server. Use `<@COL 1>` inside a `<@ROWS>` block in the Mail action's Results HTML to display these results.

The result array is also stored automatically in the local variable `resultSet`.

## Disabling Mail

For more information, see “Configuring Tango Server” on page 317.

Mail actions are enabled in Tango by default. If you want to disable (or enable) this feature, you can do so by changing the following option in the `config.taf` application file **Feature Switches** screen:

```
mailSwitch
```

Check or uncheck the check box beside the option.

CHAPTER SEVENTEEN

# Reading, Writing, and Deleting Files

17

## File Action

The *File* action allows you to read, write, and delete files on the Tango Server machine. Some of the functions you might want to perform using this action are as follows:

- Store data permanently to disk for later retrieval (as opposed to variables which are in memory).
- Keep a log file, appending to it when a certain function gets called in your application file.
- Write HTML files using database-generated data to your Web server document folder.
- Store data to a file for export to an external system, providing data navigation.
- Import data from a file from an external system, for example, daily reports from a mainframe, newswire feeds, and periodic updates to a third-party supplier.
- Use the action instead of giving FTP access to upload one or two files from a Web site.
- Use the action as an administrative tool to upload graphics or as an intranet tool to upload word processor documents or other types of documents to the server.

The topics covered in this chapter include:

- setting up a File action
- handling file security.

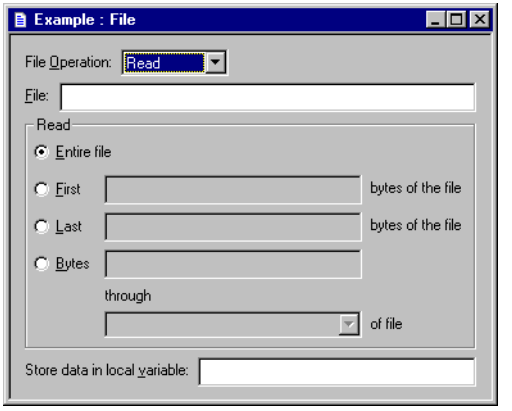
## Setting Up a File Action

You can set up the three types of file operations using the File action.

- **Read** (the default) allows you to specify the path and file name of the file you want to read. You also specify if you want to read the entire file or some part of it, and store the data in a local variable.
- **Write** allows you to specify the path and file name of the file to write data to. You also specify what that data is and, if the file exists, whether the data should be appended to or overwrite the existing data. You can also store the file name in a local variable.
- **Delete** allows you to specify the path and file name of the file you want to delete.

File Action

When you drag the File action icon from the Actions bar, the File action editing window appears.




**Note** For any of the editing fields, you can include value-returning Tango meta tags or drag snippets from the Snippets workspace. You can also right click an editing field to display a context-sensitive menu of editing commands, including the **Insert Meta Tag** command.

## Setting Up Read Options

From the **File Operation** drop-down list, select **Read**, if it is not already selected.

The action editing window changes to show the options for the **Read** type operation, which you specify as follows:

- **File**. The path and file name of the file you want to display to the user. You must specify the full, absolute path, not the path relative to the Web server root.
- **Read**. Sets which part of the file to read. You can choose **Entire file**, or select one of the following options:

  **First**. Type the number of bytes to read from the start of the file.

  **Last**. Type the number of bytes to read at the end of the file.

  **Bytes**. Type the starting and ending bytes. If the ending byte you are specifying is the end of the file, you can select **EOF** instead.
- **Store data in local variable**. The name of a local variable to store the read data in.

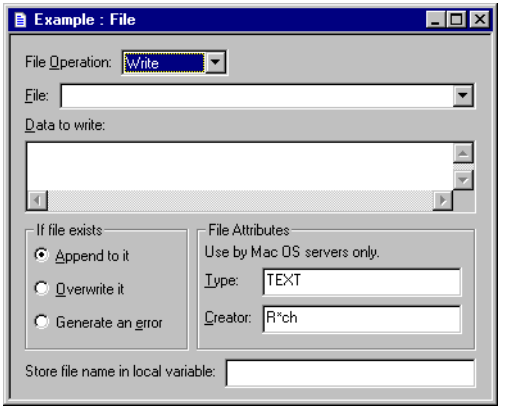
**Note** The read data is also available as `<@COL 1>` in the action's Results HTML and is stored in the local `resultSet` variable.

If you do not specify a file, the action does not do anything. In other words, the action behaves like a file exists, but the file is empty. Specifically, the specified variable is empty.

## Setting Up Write Options

From the **File Operation** drop-down list, select **Write**.

The action editing window changes to show the options for the **Write** type operation.




Specify the **Write** options as follows:

- **File**. The full, absolute path and file name of the file you want to write data to, for example:

  `c:\inetpub\wwwroot\client\uploads\mydoc.doc`

  **Note** This file information can also come from a variable or an argument.

  You can also tell Tango to generate a temporary file by selecting **Temporary file** from the drop-down list. If you select this option, the server creates a temporary file using standard routines for the operating system.

- **Data to write**. The data to write to the specified file. For example, you could enter the named post argument for a form field where the user enters the data to be saved in the specified file, or enters a file name to upload.
- **If file exists**. Specify what you want to do if the file already exists. Select one of the following options:

  **Append to it** appends to the existing file the data you are writing.

  **Overwrite it** replaces the existing data in the file with the data you are writing.

  **Generate an error** generates an error message on execution.

- **File Attributes** (used by Mac OS servers only). These Mac OS only **Type** and **Creator** codes are used when creating a file. `TEXT` and `R*ch` are the default values for new actions. They are also the values used by the server if either field evaluates to empty. The server uses the first four characters of each field (after substitution). If you specify less than four characters, the value is space padded to the end.
- **Store file name in local variable**. The name of a local variable in which to store the path and file name of the file written to. You would use this option when you write data to a temporary file.

**Note** The read data is also available as `<@COL 1>` in the action's Results HTML and is stored in the local `resultSet` variable.

## Setting Up Delete Options

From the **File Operation** drop-down list, select **Delete**.

The action editing window changes to show the options for the **Delete** type operation.


In the **File** field, specify the full, absolute path and file name of the file to delete.

## Handling File Security

For more information, see “Configuring Tango Server” on page 317.

File reads, writes and deletes are enabled in Tango by default. If you want to disable (or enable) these features, you can do so by changing the following options in the `config.taf` application file, in the **Feature Switches** screen:

```
fileReadSwitch
fileWriteSwitch
fileDeleteSwitch
```

For more information, see “Feature Switches” on page 321.

Check or uncheck the check box beside the option.

**Note** The `fileReadSwitch` configuration variable also applies to the `<@INCLUDE>` meta tag.

CHAPTER EIGHTEEN

# 18 Using Advanced Database Actions

*Setup and Operation of Transaction and Direct DBMS Actions, and Joining Database Tables*

You can put several database actions together to create a single *transaction* that manages the work being performed. Using *Begin Transaction* and *End Transaction* actions you can specify where to begin, commit, and rollback database changes.

You can use the *Direct DBMS* action to execute specified SQL statements and return any results generated.

Relational databases let you specify joins to permit searches involving more than one table. A *join* tells the database how the tables are related. A *standard join* preserves only those rows from a search in which a match exists with the joined table. An *outer join* preserves all the rows in one of the tables, even if there is no match with the other table.

The topics covered in this chapter include:

- setting up and executing Transaction actions
- setting up and executing the Direct DBMS action
- understanding joins
- creating and editing joins.

# Creating Database Transactions

Tango supports special database actions that allow you to specify where to begin, commit, and rollback database changes. Using the Begin Transaction and End Transaction actions, you can create a well-defined single transaction.

Normally, actions executed by Tango Server that change the content of databases (Insert, Update, Delete, and Direct DBMS actions) cause an immediate change to the database. This is because Tango automatically sends a `COMMIT` command as the final step in its execution of these actions.

However, Transaction actions let you control when database changes are made permanent and also let you undo (or `ROLLBACK`) the effects of actions that have been executed.

To perform a Transaction action, Tango maintains a database connection longer than it would for other actions. You should consider the impact this may have on your server and database resources before deciding to use Transactions in your application file.

**Note** Tango Transaction actions have no effect on databases that do not support transactions.

## Setting Up a Transaction Action

### Begin Transaction

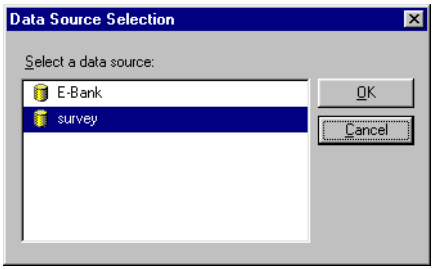
The Begin Transaction action indicates the beginning of a transaction on a particular data source.

#### *To set up a Begin Transaction action*

Begin Transaction

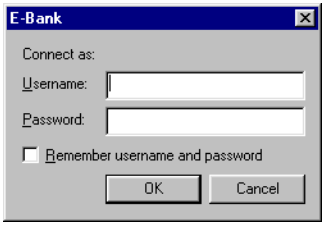
1. Drag the Begin Transaction action icon from the Actions bar into an application file.

The Data Source Selection dialog box appears.




2 Select a data source.

3 A dialog box appears where you can enter a username and password, if necessary.




Enter the Username and Password and choose **OK**.

4 Choose **OK**.

The Begin Transaction action editing window becomes active.


5 Specify the isolation level attribute you want to assign to the Begin Transaction action.

- **Read/Write exclusive**. Locks rows that are read as part of the transaction until a `COMMIT` or `ROLLBACK` command is issued to the database server.
- **Read uncommitted**. Reads rows that have been changed by other database users in a transaction, but for which the transaction has not been committed or rolled back.

## End Transaction

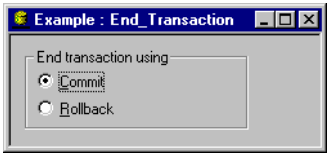
The End Transaction action marks the end of the transaction and either commits it (saves all the changes) or rolls it back (discards all the changes).

### *To set up an End Transaction action*

End Transaction

1 Drag the End Transaction action icon from the Actions bar into an application file.

The End Transaction action editing window appears.




2 Specify in the action editing window the attribute you want to assign to the End Transaction action:

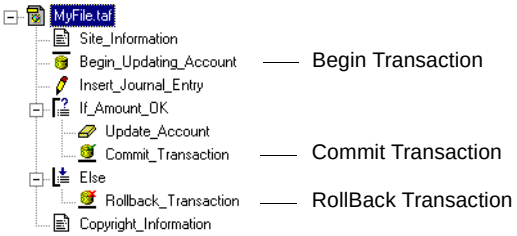
Commit End Transaction

RollBack End Transaction

- **Commit**. Commits any modifications made to the database during the current transaction.
- **Rollback**. Undoes any modifications made to the database during the transaction.

In the application file, the End Transaction action icon changes to reflect the associated attribute.

The following is an example of valid Transaction actions appearing in an application file.



## Executing a Transaction Action

If Tango Server detects that an End Transaction action is being executed without first executing a Begin Transaction action, it reports a runtime error. It is also an error to begin another transaction before an existing transaction is committed or rolled back.

Database actions on data sources that are not the transaction data source are automatically committed.

If the application file ends without executing an associated End Transaction action or a Return action, then a Rollback End Transaction action executes automatically.

**Tip** When executing a transaction, your application could slow down; additional RAM may be required for Tango Server.

# Executing SQL

The Direct DBMS action executes specified SQL statements and returns any results generated.

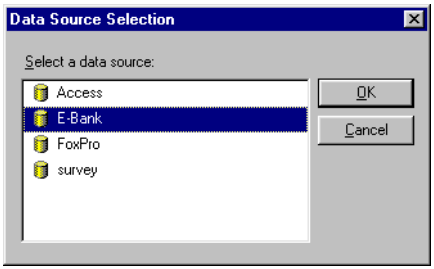
## Setting Up a Direct DBMS Action

Direct DBMS Action

### *To set up a Direct DBMS action*

1. Drag the Direct DBMS action icon from the Actions bar into an application file.

   The Data Source Selection dialog box appears.

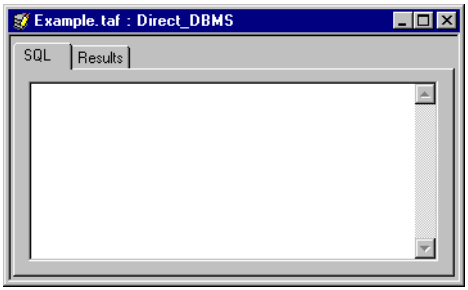



2. Select a data source.
3. A dialog box appears where you can enter a username and password, if necessary.


   Enter the Username and Password and choose **OK**.

4 Choose **OK**.

An empty Direct DBMS action editing window appears, displaying SQL and Results tabs.




Fill in the tabbed windows as described next.

## The Direct DBMS Action Editing Window

### SQL Tab

Click the SQL tab to display the Direct DBMS action SQL text area in which to enter the SQL to be executed.

All statements are executed against the database specified in the data source associated with the Direct DBMS action.

For information on constructing SQL statements, consult your database or ODBC driver documentation.

You can easily enter column or table names by dragging them from the Data Sources Workspace. If you drag multiple columns, they are separated with commas.

You can also perform standard editing operations in the Direct DBMS action editing window. Editing commands are available from the **Edit** menu.

You can also right click the Direct DBMS action editing window to display a context-sensitive menu of commands.

You can reference any value-returning Tango meta tags in your SQL.

#### *To insert a meta tag in the Direct DBMS window*

1 Click the action editing window where you want to enter a meta tag.

2 Do either of the following.

- From the **Edit** menu, choose **Insert Meta Tag**.
- Right click the action editing window, and choose **Insert Meta Tag** from the context-sensitive menu that appears.

The Insert Meta Tag dialog box appears allowing you to specify a meta tag and inserts it at the insertion point in the SQL text area.

For information on filling in the Insert Meta Tag dialog box, see “Inserting Meta Tags” on page 113.

For an example of executing different SQL based on the type of data source in use, see "<@DSNUM>" on page 77 of the *Meta Tags and Configuration Variables* manual.

You can use the <@IF>, <@IFEQUAL>, and <@IFEMPTY> meta tags in the SQL text to include or exclude SQL based on the result of a comparison at execution time. For example, you could use this capability to execute different SQL based on the type of data source in use.

## Direct DBMS SQL Auto-Encoding

For more information, see "SQL" on page 10 of the *Meta Tags and Configuration Variables* manual.

Tango Server automatically performs SQL encoding on meta tag values substituted in Direct DBMS SQL. For example, if a variable called myName contains "O'Brien":

```
SELECT * FROM customer WHERE cust_name = '<@VAR
NAME=myName>'
```

This results in:

```
SELECT * FROM customer WHERE cust_name = 'O''Brien'
```

If Tango did not do this, the result would be:

```
SELECT * FROM customer WHERE cust_name = 'O'Brien'
```

and a DBMS error would result due to the unescaped quote.

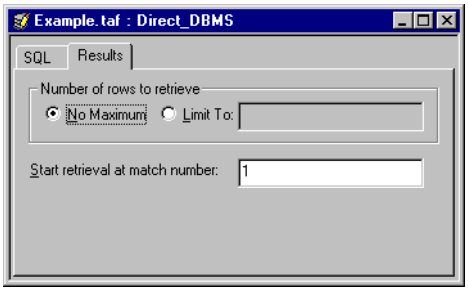
For more information, see "Encoding Attribute" on page 9 of the *Meta Tags and Configuration Variables* manual.

If your Direct DBMS SQL contains meta tags that evaluate to an entire (or partial) SQL statement constructed elsewhere, the quote-doubling may cause DBMS errors. This is because all single quotes are doubled, even those meant to delimit a string. In this case, the solution is to modify the meta tag(s) returning your SQL by adding the ENCODING=none attribute. For example:

```
<@VAR NAME=mySQL ENCODING=none>
```

## Results Tab

Click the Results tab to display the results options you can set for the Direct DBMS action.



You can specify options for the maximum number of records to retrieve from the data source and at which result record number retrieval begins.

### *Number of rows to retrieve*

To return all matching rows, select **No Maximum**.

To limit how many records are returned by the action, select **Limit To** and enter the maximum number of rows to retrieve.

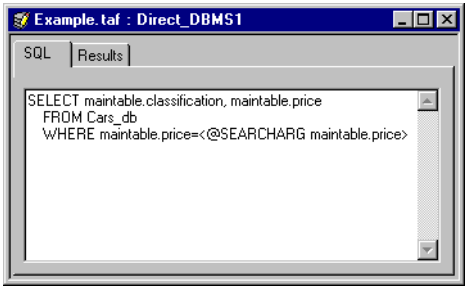
### *Start retrieval at match number*

Use this option to skip some of the matching records. Enter “1” (the default) to start retrieval with the first matching record. When a value other than “1” is entered into this field, the Direct DBMS action returns records starting at that number, skipping any records before it.

Both of these fields can contain meta tags which return values.

## Executing a Direct DBMS Action

When Tango Server executes a Direct DBMS action, the specified SQL is sent to the data source for execution.




Any result rows are returned to Tango and may be accessed in the action’s Results HTML. As with the Search action, a `<@ROWS>` `<@/ROWS>` block is used to iterate through the records returned. You must specify column references differently, however.

You can use `<@COLUMN>` to refer to your columns by name for ODBC data sources, but for non-ODBC data sources, you must refer to your columns in Results HTML by number, using the `<@COL>` meta tag.

For example, if your Direct DBMS action executed the following statements with a non-ODBC data source:

```
SELECT maintable.price, maintable.classification,
maintable.manufacturer
       FROM maintable
```

the following Results HTML would print the database results:

```
<@ROWS>
maintable.price: <@COL 1><BR>
maintable.classification: <@COL 2><BR>
maintable.manufacturer: <@COL 3><BR>
<HR>
</@ROWS>
```

If the Direct DBMS action generates no database results, and you have specified No Results HTML for the action, that HTML is processed instead of the Results HTML.

**Notes**

- The Tango Oracle data source type does not permit the retrieval of results from a procedure execution specified in a Direct DBMS action. Only `SELECT`'s return data for Oracle data sources.
- When calling a stored SQL server procedure from a Direct DBMS action with an ODBC data source, you should use the following syntax:

  `{call procedureName(`*param1*`,`*param2*`,`*paramX*`)}`

# Joining Database Tables

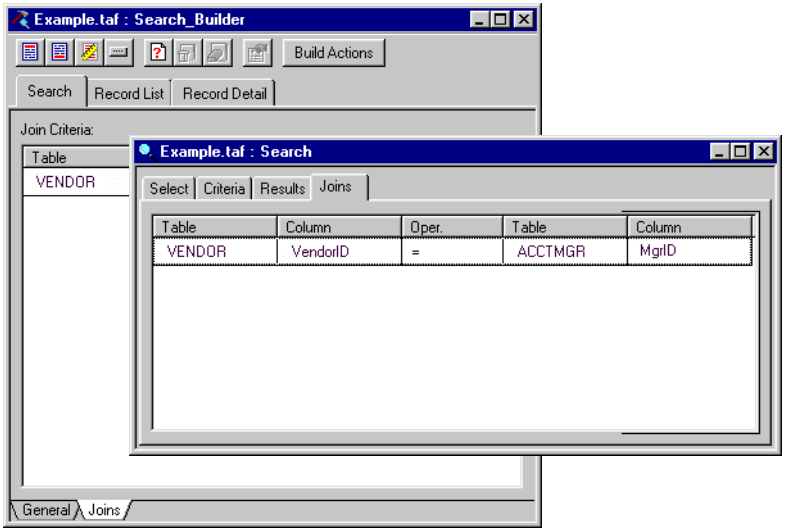
For more information on joins, consult your DBMS documentation, such as, *The Practical SQL Handbook* (J.S. Bowman, *et. al.*, ISBN: 0-201-62623-3), or any other good SQL reference guide.

To understand how joins work, consider a database with vendor and associated account manager information in two different tables. You want to create a search to find the account manager for any given vendor, and display in your browser the vendor information with the corresponding account manager's name. Because the vendor table contains only the account manager's identifier, you have to *join* the two tables to get the account manager's full name.

The vendor table (VENDOR) has a record for each vendor including a vendor identifier, name, contact information, payment terms, and an account manager identifier. The account manager table (ACCTMGR) has a record for each account manager including the account manager's identifier, name, and telephone number. The vendor table is related to the account manager table by an identifier in the AcctMgr column; the account manager table has a MgrID column that contains the identifier corresponding to the AcctMgr column in the vendor table.

For more information, see "Creating a Join in a Search Action" on page 298.

Using a Tango search, you select the columns you want to relate and define the type of join in the Joins tab of the Search action or Search Builder.



In addition to the standard join, you can define an *outer join*, which can be left or right. A *left outer join* means all rows in the left-specified table are returned, including those with no match in the right-specified table. A *right outer join* means all rows in the right-specified table are returned, including those with no match in the left-specified table.

For this example, you would select the MgrID column in the left table, ACCTMGR, and the AcctMgr column in the right table, VENDOR. Then, from the drop-down list you select the type of join you want to use.

- If you select a *standard join* (=), the search returns *only* rows of vendor information where a valid account manager's identifier is found. If none is found, the corresponding row is not returned. For instance, if a vendor has not been assigned an account manager and thus the MgrID column is blank for that record, that vendor is not returned.

| Vendor ID | Name | Account Manager | City | State |
|---|---|---|---|---|
| 1 | ACME Accessories | Erin Shanahan | Groton | MA |
| 2 | Norris Corporation | Andreas Franck | Groton | MA |
| 4 | Smith & Waterman Assoc. | Wendell Ainsworth | Foster City | CA |
| 5 | West End Corp. | Paula Jason | Cheylan | WV |
| 6 | Worldwide Posters Unlimited | Andreas Franck | Eugene | OR |

Only rows with matching account managers are returned.

- In contrast, if you define a *right outer join* (=*), the search returns *all* rows of vendor information, regardless of whether an account manager is found or not.

| Vendor ID | Name | Account Manager | City | State |
|---|---|---|---|---|
| 1 | ACME Accessories | Erin Shanahan | Groton | MA |
| 2 | Norris Corporation | Andreas Franck | Groton | MA |
| 3 | PrintCo | | Carson City | NV |
| 4 | Smith & Waterman Assoc. | Wendell Ainsworth | Foster City | CA |
| 5 | West End Corp. | Paula Jason | Cheylan | WV |
| 6 | Worldwide Posters Unlimited | Andreas Franck | Eugene | OR |

In this case, the row is returned even though no matching account manager is found.

- A *left outer join* (*=) returns rows of vendor information based on the account managers found, including any account managers without vendors assigned.

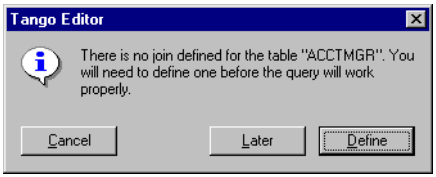
| Vendor ID | Name | Account Manager | City | State |
|---|---|---|---|---|
| 0 | | Jamie Frame | | |
| 1 | ACME Accessories | Erin Shanahan | Groton | MA |
| 2 | Norris Corporation | Andreas Franck | Groton | MA |
| 4 | Smith & Waterman Assoc. | Wendell Ainsworth | Foster City | CA |
| 5 | West End Corp. | Paula Jason | Cheylan | WV |
| 6 | Worldwide Posters Unlimited | Andreas Franck | Eugene | OR |

In this case, the row is returned even though the account manager found has no matching vendor.

## Working With Joins

To work with joins, you must first have your Search action or Search Builder action editing window open.

You can include columns from more than one table in a search, if you define joins for the tables. If you select columns from more than one table in a search, a message appears telling you to define a join.




Choose either **Define** to create the join definition now or **Later** if you want to define the join at a later time. Choosing **Define** opens the Joins window of the current dialog box.

You create, modify, and delete joins using the Joins tab of the Search editing window or Search Builder.

When you define the join, it adds the columns to the search. In the Search Builder, you must define the join before you build the actions for the search or save the application file.

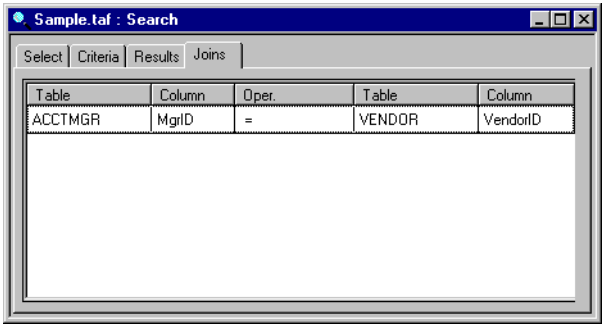
**Note** In earlier versions of Tango you could get join information by using the **Attribute** menu's **Joins** command or the Joins icon in the Attributes bar. Join information is viewed in the Search action or Search Builder editing window.

## Creating a Join in a Search Action

***To create a new join in a Search action***

**1** In the Search editing window, click the Joins tab.

The Joins window opens.




If you added columns from different tables to the **Select Columns** list (under the Select tab), a join definition already appears, showing you the tables selected and the first column added from each table. The default operator is "=".

If you have not added columns, do so now by dragging the columns into the Joins window.

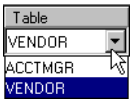
**Note** You cannot create a join from two different data sources.

**2** From the **Table** drop down lists, select the left and right tables for the join.

To select an entry for any definition field, do either of the following:

- Click the field twice.
- Right click and choose **Edit** from the context sensitive menu.

The field changes to a drop down list box so you can choose a different entry.

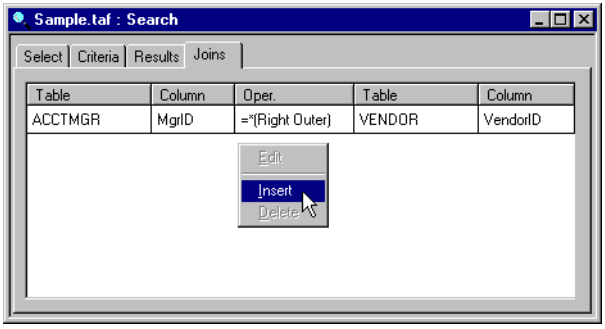



**3** From the **Column** lists, select the columns you want to join in each table. A table's first column appears as the default in the list.

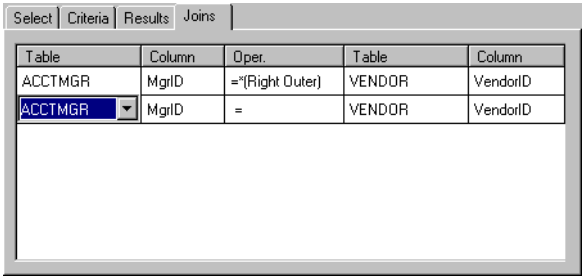
**4** From the **Oper.** drop down list, select a join operator.

| Join Operator | Description |
|---|---|
| = | Standard join (the default). Only records matching the join criterion are returned. |
| *= | Left outer join. All left-table rows are returned, including those with no match in the right table. |
| =* | Right outer join. All right-table rows are returned, including those with no match in the left table. |

To add additional join definitions, right click the Joins window and choose **Insert** from the context sensitive menu.




A new row is added to the window.



## Editing a Join

***To edit an existing join definition***

1. Click the Joins tab in the Search action editing window.

   The Joins window appears, showing you the current join definition(s) including table names, joined columns, and join operator.

2. Do either of the following to edit a definition field:

   - Click the field twice.
   - Right click and choose **Edit** from the context sensitive menu.

   The field changes to a drop down list box so you can choose a different entry.

## Deleting a Join

***To delete an existing join definition***

1. Click the Joins tab in the Search action editing window.

   The Joins window appears, showing you the current join definition(s) including table names, joined columns, and join operator.

2. Do either of the following to delete a join definition or definitions from the Joins window:

   You can SHIFT-click (contiguous rows) or CTRL-click (discontiguous rows) to select multiple join definitions.

   - Select the join definition(s) and press DELETE.
   - Right click the selected join definition(s) and choose **Delete** from the context sensitive menu.

   A message appears asking you to confirm that you want to delete the selected row(s).

3. Click **Yes** to delete the selected rows or **No** to cancel.

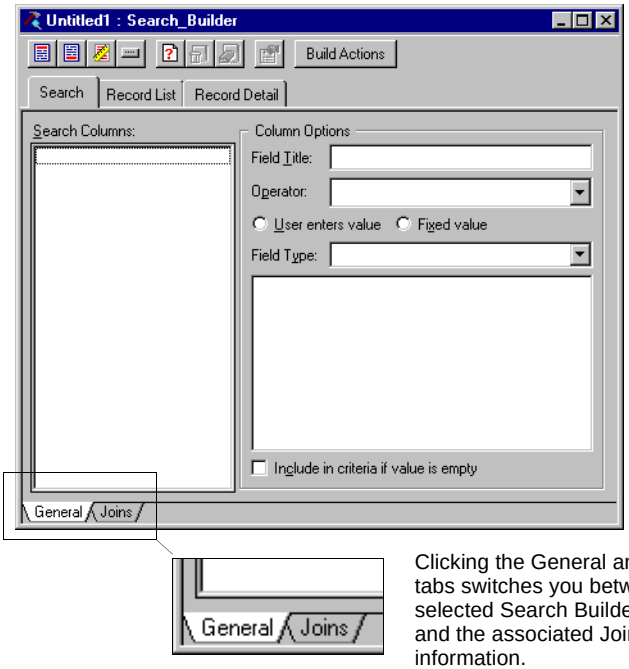
**Note** If your Search action refers to columns from the deleted joined table, you need to remove these columns and references from the action or builder window manually.

## Creating a Join in the Search Builder

You create, edit, and delete joins in the Joins window the same way you do for a Search action. See “Creating a Join in a Search Action” on page 298.

The Search and Record List option groups of the Search Builder share the same join information because they both apply to the same generated Search action. You can specify separate join information for the Record Detail option group.

You switch from the Search, Record List, or Record Detail window to the associated Joins window by clicking the General and Joins tabs, respectively, at the bottom of the main Search Builder window.




Clicking the General and Joins tabs switches you between the selected Search Builder tab and the associated Joins information.

**Note** You cannot create a join between two different data sources.

CHAPTER NINETEEN

# Setting Preferences

19

## Changing Your Tango Editor Preferences

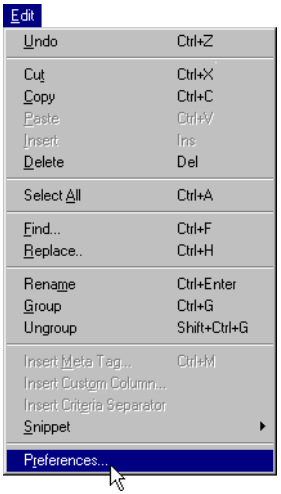
The default preferences for Tango Editor are automatically set during installation. If you want to change the various settings required by the application, you can do so using the Preferences dialog box.

This chapter describes each of the following preference settings:

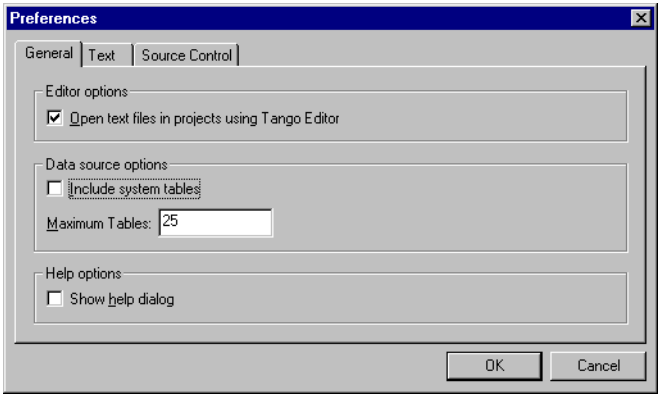
- editor, data source, and online help dialog box options
- HTML and text options
- source control options.

# Using the Preferences Dialog Box

From the **Edit** menu, choose **Preferences**.




The Preferences dialog box appears.



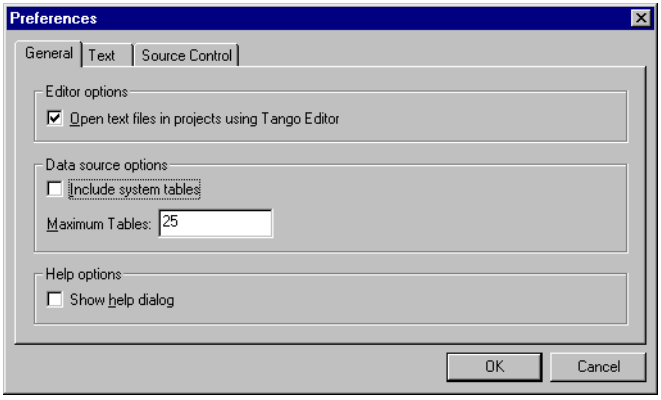

You set Tango preferences using three tabs: the first for general preferences, the second for how you want the text to look in Tango editing windows, and the third for source control. You switch among preference sections by clicking the appropriate tab to display the options available.

After you set your preferences, clicking **OK** saves your changes and closes the Preferences dialog box. Any open editing windows are automatically updated with any new settings.

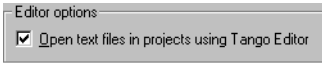
# Selecting Options

## General

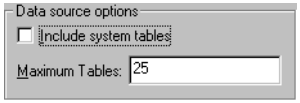
To display the General section of the Preferences dialog box, if not already displayed, click the General tab.




- **Editor options**. Opening a text or HTML file in the Project Workspace (that is, a file with any of the extensions listed in the table on page 64) by default opens the file in Tango's built-in editing window. If you want instead to use the application defined for that file type in the Windows Explorer, disable the **Open text files in projects using Tango Editor** option.




- **Data source options**. Select the **Include system tables** option to include the data source's system tables in the Data Sources Workspace. This option is disabled by default.



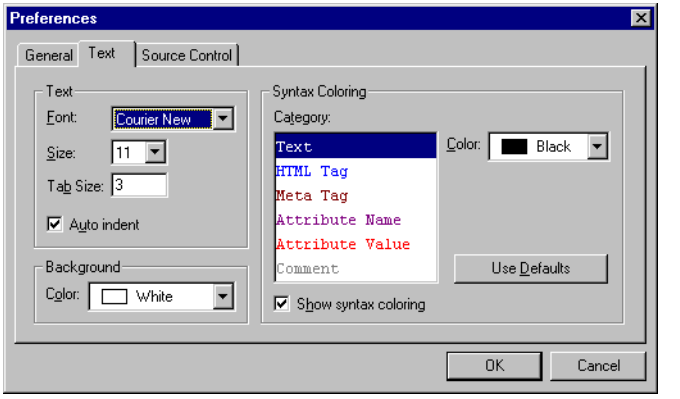

  Set the maximum number of tables you want to appear. The default is "25". If a data source has more than the specified number of tables, the Select Tables dialog box appears, allowing you to work with a more manageable subset of tables.

- **Help options**. Select the **Show help dialog** option to show the Help Information dialog box, which tells you about the HTML help system when you choose an item from the **Help** menu.

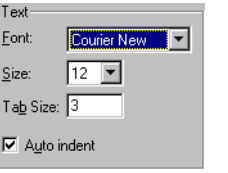
Selecting this option is the only way to show the help dialog again if you have previously selected **Don't show this dialog again** in the Help Information dialog box.

## Text

When you click the Text tab in the Preferences dialog box, the following text options appear.




### Text




Select the basic text attributes for the text that appears in the editing windows.

- **Font**. Select from the monospaced fonts installed on your machine, such as, Terminal, Fixdsys, Courier, Courier New, and Lucida Console.
- **Size**. Select from the point sizes available for the selected font, such as, 9, 10, 11, 12, 14, 16.
- **Tab Size**. Type the number of characters you want to equal one tab character. The default is "3".
- **Auto indent**. This option, enabled by default, inserts a tab character automatically at the start of a new line at the same indent level as the previous line.

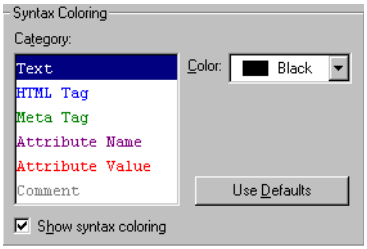
## Background


**Color** refers to the background color of the HTML editing window.

You can select from the colors in the drop down list: Black, Maroon, Green, Olive, Navy, Purple, Teal, Gray, Silver, Red, Lime, Yellow, Blue, Fuschia, Aqua, and White. The default is “White”.

If you select a different color, the background of the **Category** list changes to show you the selected font, size, and color against that background.

## Syntax Coloring




In addition to setting a default font for text appearing in the HTML editing window, you can also add color to the selected font for certain categories of text.

Coloring your text can make editing of your text, HTML, and meta tags much faster and easier, and reduce the chances of making syntax errors. Only *valid* HTML and meta tags appear in the specified color.

**Note** Meta tag attribute names are not currently checked for validity.

The default is to show the editing window text with syntax coloring enabled. If you disable the **Show syntax coloring** option, all text in an editing window appears black on a white background.

The following table describes each category and the default color for the text in the category.

| Category | Text Affected by the Setting | Default Color |
|---|---|---|
| Text | Text that is neither a meta tag nor HTML. | Black |
| HTML Tag | HTML tag names, for example, <BODY> and </BODY> | Blue |
| Meta Tag | Meta tag names without any attributes, for example, <@POSTARG> | Green |
| Attribute Name | Meta tag attribute name, for example, NAME= in <@POSTARG NAME="Fred"> | Purple |
| Attribute Values | Meta tag attribute value, for example, "Fred" in <@POSTARG NAME="Fred"> | Red |
| Comment | Any text enclosed within the <@COMMENT> <@/COMMENT> meta tag pair, including the <@COMMENT><@/COMMENT> meta tags. This category also includes the <@!> meta tag and HTML comments. | Gray |

To assign a different color to a category, select the category in the list, then select a color from the **Color** drop down list. The colors available are the same colors listed for “Background” on page 307.

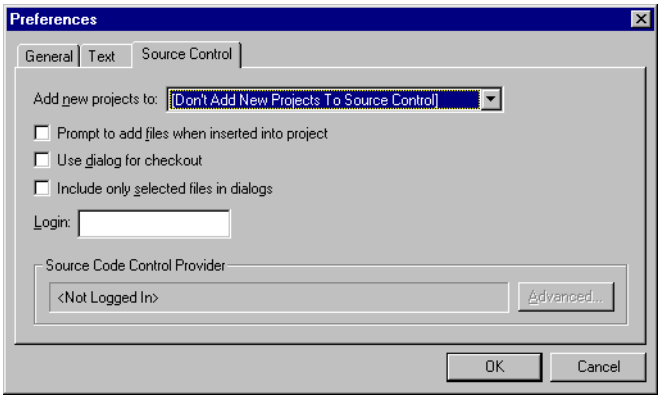
If you change text preferences and want to return to the defaults, click **Use Defaults**.

## Source Control

When you click the Source Control tab in the Preferences dialog box, the following source control options appear.

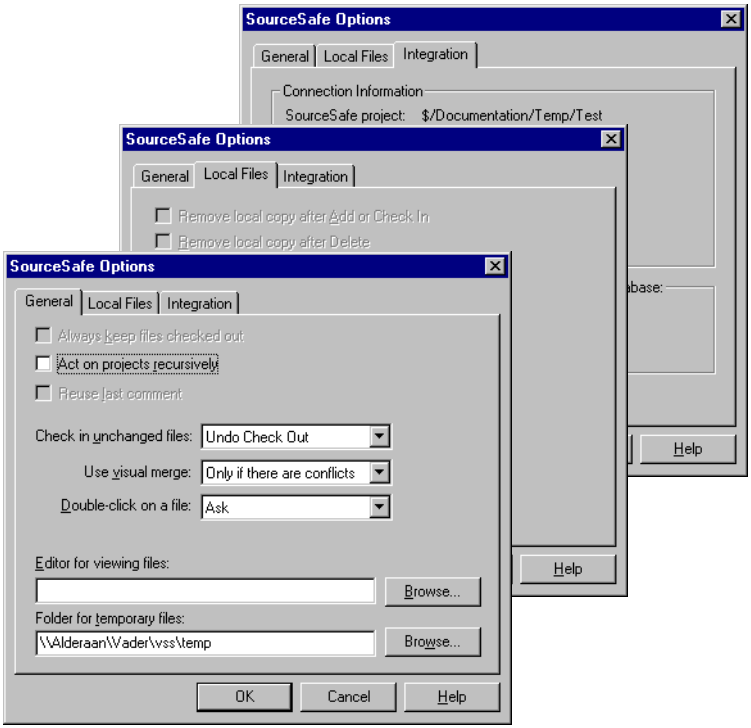
For more information, see “Using Source Control in Tango” on page 67.

**Note** The Source Control tab only appears if you have a source control system installed on your machine. You must have your source control system’s client software installed on the same machine as Tango Editor.




- **Add new projects to**. To add newly created Tango projects automatically to source control, select the name of your source control system in the drop down list. The default is **Don’t Add New Projects To Source Control**.
- **Prompt to add files when inserted into project**. If you enable this option, when you add new files to a Tango project, Tango Editor asks if you want to add the new files to source control. Click **Yes** to add the files or **No** to cancel.
- **Use dialog for checkout**. Enable this option if you want the Check Out File(s) dialog box to appear always when you check out files. Otherwise, the selected files are automatically checked out without displaying the dialog box. A checkmark in the checkbox (☑) beside the file name in the Project Workspace indicates the file is checked out.
- **Include only selected files in dialogs**. Enable this option if you want the Get Latest Version, Check Out File(s), Check In File(s), and Undo Check Out dialog boxes to show only the files selected in the Project Workspace for the particular operation. If disabled, you do not see all the files the particular source control operation could apply to.

- **Login**. Enter the name you use to log in to your source control system. The name you enter automatically appears in your source control system's login dialog box; otherwise, the user name field remains empty.
- **Source Code Control Provider**. This area displays the name of your source control system. If you are not logged into your source control system, the area is grayed out and contains the message "not logged in".
- **Advanced**. Click this button to display some of the options available for your source control system. Because the options appearing correspond to the options for your particular source control system, they may appear differently than this example shows.




Click **Help** for a description of each of the available advanced options and how to set them.

CHAPTER TWENTY

# Using Tango Server

20

## Changing Tango Server Defaults

As discussed in Chapter 1, the program that queries the database and returns information to Web browsers is *Tango Server*, which is the main component of Tango. It is responsible for executing application files, communicating with databases directly or through ODBC, and compiling the HTML to be returned to the Web server.

For information on how to install Tango Server, see the *Getting Started Guide* that comes with Tango.

Also included in the Tango package is a CGI program, a small program that uses the Common Gateway Interface (`t3.cgi` under UNIX, `t3cgi.exe` under Windows) and routes requests between Tango Server and the interface to the database that you have configured. This CGI routing of information can also be done with a Web server plug-in for Netscape Server or Microsoft Internet Information Server. This chapter also describes the defaults for the plug-in or CGI program.

Some settings for Tango Server are kept in a configuration file (`t3server.ini` under Windows). This chapter describes how to change the settings in this file using the `config.taf` application file and discusses the defaults that are set when Tango is installed. Because of the many cross references in this chapter, references to the *Meta Tags and Configuration Variables* manual are indicated with *MTCV*.

This chapter describes:

- timed URL processing with Tango
- startup and shutdown query processing
- configuration file defaults for Tango Server and the CGI program and modifying them with the `config.taf` application file.

# Timed URL Processing With Tango Server

The timed URL execution (*cron*) component of Tango Server lets you execute specified URLs at stipulated time intervals. Only HTTP-type URLs are supported. The URLs may point to the local Web and Tango server or a remote one. (The result HTML of an executed URL is discarded.)

Both the specification file and the execution mechanism are modeled after the UNIX `cron(1)` daemon. The specifications are stored in a separate file (conventionally called `crontab`), which is read when Tango Server starts up. You must restart Tango Server if the file is modified.

The timed query module of Tango Server becomes active once per minute, and executes, one by one, all the URLs that satisfy the specification conditions between the previous and present activation.

For more information, see “crontabFile” on page 169 of the *Meta Tags and Configuration Variables* manual.

The full path to the crontab file must be specified in the system configuration variable `crontabFile`. The following section discusses the format of the `crontab` file.

## Format of the crontab File

A `crontab` file consists of lines of six fields each. The fields are separated by spaces or tabs. The first five fields are numeric patterns and specify the following (in this order):

- minute (0–59)
- hour (0–23)
- day of the month (1–31)
- month of the year (1–12)
- day of the week (0–6 with 0=Sunday)

Each of these patterns may be either an asterisk (meaning all legal values) or a list of elements separated by commas. An element is either a number or two numbers separated by a dash (meaning an inclusive range). The specification of days may be made by two fields (day of the month and day of the week), or both.

The sixth field of a line in a `crontab` file is a URL executed by Tango Server at the specified time(s). It must be in HTTP-type URL form, which is:

```
http://host:port/path?search-arguments
```

*Host* may be specified as fully qualified hostname or Internet address, and may not be omitted. *Port* may be omitted, in which case the default HTTP port (80) is assumed. *Path* and *search arguments* may be omitted in which case a request for the main page is sent.

Lines starting with hash sign (#) are treated as comments. The following is an example of a crontab file.

```
#
# Example crontab file
#
# Run this application file all the time (every minute)
* * * * * http://127.0.0.1/cgi-bin/t3.cgi/query/
null.taf
# Gather statistics every hour, Monday through Friday,
# 10am to 6pm
1 10-18 * * 1-5 http://127.0.0.1/cgi-bin/t3.cgi/query/
stats.taf
# Run this query every minute, Monday through Friday,
# 10am to 6pm
* 10-18 * * 1-5 http://127.0.0.1/cgi-bin/t3.cgi/query/
stats.taf
# Clean up database locks every Sunday =and= on 1st and
```

```
# 15th of the month
0 1 1,15 * 0 http://127.0.0.1/cgi-bin/t3.cgi/query/
dbclean.taf
# Run this application file at noon on 14th of July
0 12 14 7 * http://127.0.0.1/cgi-bin/t3.cgi/query/14-
Juillet.taf
```

## Startup and Shutdown URL Processing

Startup and shutdown URLs get executed by Tango Server on initialization and shutdown. This mechanism is provided for the purpose of initialization and finalization of the Tango Server environment.

These URLs can point to application files, which can, for example, set up some installation-specific system variables on startup and gather and mail statistical data on shutdown. These URLs can point to CGI scripts that start or stop database engines, or to change run levels on some of the intranet computers. They can also be remote requests to obtain some vital information from other Internet sites. The Results HTML of startup and shutdown URLs are discarded.

### Specifications

Startup and shutdown URLs are specified in system configuration variables (`startupURL`, `shutdownURL`, and `startstopTimeout` ). Tango Server must be restarted for new values to take effect.

For more information, see “shutdownUrl” on page 185, “startStopTimeout” on page 185, and “startupUrl” on page 186 of the *Meta Tags and Configuration Variables* manual.

`startupUrl` specifies the URL which is executed on Tango Server startup. `shutdownUrl` specifies the URL which is executed when Tango Server receives a shutdown request. `startstopTimeout` specifies, in seconds, the time which Tango Server is allowed to wait for startup or shutdown URL to complete. By default, the waiting period is 60 seconds.

For a description of the URL format, see page 313.

Startup and shutdown URLs must be compliant with HTTP type URL specifications, which is, in simplified form:

```
http://host:port/path?search-arguments
```

### Examples

Application file, executed by Tango itself

```
http://www.myhost.ca/cgi-bin/MyTango.cgi/
sys.taf?action=start
http://www.myhost.ca/cgi-bin/MyTango.cgi/
sys.taf?action=stop
```

Intranet URL to start the database

```
http://dbhost.myorg.ca/cgi-bin/startdb.sh
```

Remote URL

```
http://www.remote-tango-server.com/sys.taf?
remote-start
```

## Execution Mechanism

A shutdown signal is sent to Tango Server by typing `tangod -k` on the UNIX command line, pressing STOP in service manager under Windows NT, or typing CTRL-C when Tango Server is running as a non-detached application on UNIX, Windows NT or Windows 95.

The life cycle of Tango Server consists of the following stages:

- **Born**. Tango Server performs internal initialization and moves to the next state. No user requests are accepted.
- **Start up**. Tango Server looks for the startup URL. If this URL is specified, the request is sent. If, for any reason, the request cannot be sent, or startup URL is not specified, Tango Server immediately moves to state *Running*. If the request is sent out successfully, Tango Server moves to the next state. User requests are not accepted.
- **Receiving response**. Tango Server stays in this state, until the response to the startup query arrives, or, the time elapsed after the request was sent exceeds the specified timeout. In both cases, Tango Server moves to the next state. If neither is true, Tango Server stays in the current state. User requests are still not accepted.
- **Running**. User requests are accepted. If a shutdown signal is received, Tango Server moves to the next state.
- **Shutdown**. User requests are no longer accepted. Any existing user requests are still allowed to run. Tango Server looks for a shutdown URL. If a URL is specified, the request is sent. If, for any reason, the request cannot be sent, or a shutdown URL is not specified, Tango Server immediately moves to state *Dead*. If the request is successfully sent, Tango Server moves to the next state.
- **Receiving response**. Tango Server stays in this state, until the response to the shutdown URL arrives, or the time elapsed after the request was sent exceeds the specified timeout. In both cases, Tango Server moves to the next state. If neither is true,

Tango Server stays in the current state. User requests are not accepted, but any existing user requests are still allowed to run.

- **Waiting for running threads.** Tango Server waits for still-running threads, but no longer than a specified thread timeout.
- **Dead**. Tango Server stops and kills still running threads, performs internal clean-up and exits.

## Guaranteed Execution and Guaranteed Shutdown

There could be circumstances when a malformed application file could prevent Tango Server from shutting down in a timely fashion.

When Tango Server is running under UNIX as a detached process (daemon) or on a terminal, or as a console application on Windows NT or Windows 95, a second shutdown signal causes the application to immediately abort, not depending on its state. When run as NT service, the internal mechanisms are employed to guarantee a timely shutdown because a second shutdown signal cannot be delivered.

If a shutdown signal arrives before Tango Server has finished initialization, (that is, before it has moved to *Running* state), the request is honored only after the initialization stage is complete, and, after that, processed according to the described mechanism.

Thus, startup and shutdown URLs are always guaranteed to be fully executed without overlaps. It is also guaranteed that no user request is accepted before the initialization is completed, or after the first shutdown signal is received by Tango Server.

## Limitations

The startup and shutdown timeout works properly only after the request has been successfully sent. There is still a possibility of a large delay during DNS lookup or TCP/IP handshake.

# Configuring Tango Server

The Tango Server preferences file, `t3server.ini` under UNIX and Windows (`Tango 3 Server Preferences` under Mac OS), contains configuration settings for many Tango Server options. `t3server.ini` is located in within the `Tango3` folder in the `Windows` or `Winnt` folder on the hard drive that contains the system software. Configuration options are set by default. You generally do not need to edit the preferences file directly.

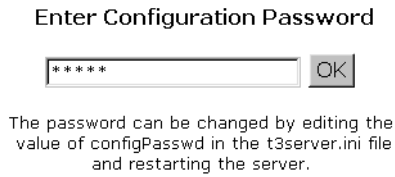
You configure Tango Server by setting system configuration variables using the `config.taf` application file that comes with Tango.

This file is located in the `\Tango3\Config\` directory in your Web server root folder.

To use it, execute the file in your browser, for example,

`http://`*`www.yourserver.com`*`/Tango3/Config/config.taf`

When you execute the `config.taf` application file, you are asked for a password.

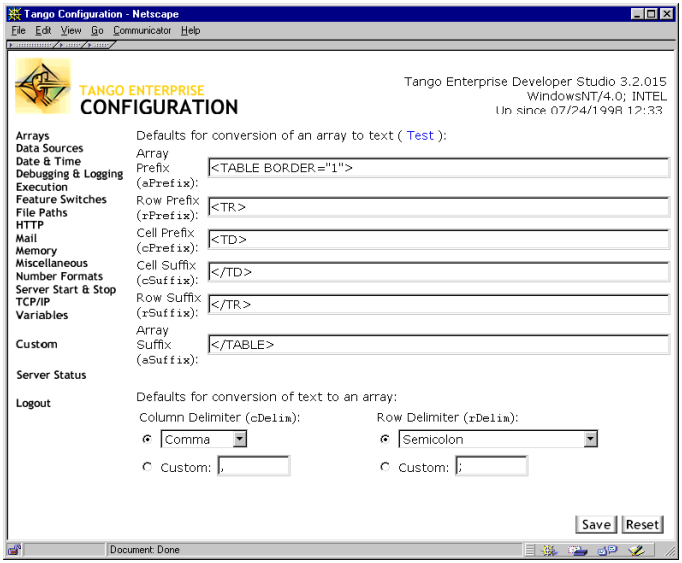



The default password after installation is the first five characters of your box number. You can change the password by editing the value of the configuration variable, `CONFIGPASSWD`, in the configuration file. This change takes place when you restart Tango Server.

**Note** If you are editing the configuration file, you may see file stanzas for more than one server. This can occur because the same configuration file could be used by multiple servers. For example, one machine could be using Tango Personal Server (Development Studio) for development and a second remote machine could be using Tango Server for deployment. If you are editing the configuration file make sure you modify the correct file stanza, that is, `[Tango_3_Personal_Server]` or `[Tango_3_Server]`.

For detailed information on configuring Tango Server, see Chapter 7 in the *User's Guide*, and the *Meta Tags and Configuration Variables* manual. You can also get configuration information by clicking the **Tango Server** button on the on-line HTML help button bar.

When the `config.taf` file loads into your Web browser, click a category on the left to see the options that can be set for that category. The listed options correspond to Tango system configuration variables.




**Caution** The configuration options affect basic Tango Server functionality. Before changing any value, you should understand the effect of the change. See Chapter 2, “Configuration Variables,” in the *Meta Tags and Configuration Variables* manual for details.

To save any changes that you have made, click **Save**. To reset the values to what they were previously, click **Reset**.

You can click on **Server Status** to get a list of current statistics about the running copy of Tango Server.l

The following describes the screens in the `config.taf` file which affect system configuration variables. Some configuration variables are found on more than one screen, because they fit into more than one category of system configuration variable.

## Arrays

The options in this screen set the way that array variables are formatted when returned in HTML.

`aPrefix` (default value `<TABLE>`)
`aSuffix` (default value `</TABLE BORDER="1">`)
`rPrefix` (default value `<TR>`)
`rSuffix` (default value `</TR>`)
`cPrefix` (default value `<TD>`)
`cSuffix` (default value `</TD>`)

For more information, see “Arrays” on page 126.

These options set the prefixes and suffixes for when Tango returns values for arrays with the `<@VAR>` meta tag (when arrays are converted to text; for example, in Results HTML).

`cDelim`
`rDelim`

For more information, see “Arrays” on page 126. See also *MTCV*, page 32, page 167, and page 184.

These options set the delimiters for array values when setting the value of arrays with the `<@ARRAY>` tag. You can select a value from the drop-down menu or enter a custom value. The default values are “,” and “;”, respectively. (These delimiters can be changed for a particular instance of an array assignment using the attributes of the `<@ARRAY>` tag.)

## Data Sources

The options in this screen set the defaults for connecting to data sources in Tango Server.

`dataSourceLife`

See also *MTCV*, page 170.

This option indicates how long the Tango Server keeps open an unused connection to a data source. You can select a value from the drop-down menu or enter a custom value. The default value is 30 minutes.

`itemBufferSize`

See also *MTCV*, page 177.

This option sets the maximum column length in bytes that Tango supports. You can select a value from the drop-down menu or enter a custom value. The default value is 64K.

`dbDecimalChar`

See also *MTCV*, page 173.

This option tells Tango Server what decimal character ODBC data sources require in numbers. This value may be determined by the ODBC driver, the database vendor’s client software, or the DBMS server. The default value is a period “.”.

`stripCHARs`

See also *MTCV*, page 186.

This option sets whether CHAR (fixed-length text field) data from data sources is automatically stripped of trailing spaces. The default value of this variable is enabled (`true`).

`transactionBlocking`

See the *Meta Tags and Configuration Variables Addendum*.

This option sets whether other processes are blocked during database transactions. The default value is true (checked).

**Caution** Setting this configuration variable to false (unchecked) may cause poor performance due to record contention when multiple users are executing transactions with Tango.

## Date and Time

The options in this screen set the defaults for the way that dates and times are formatted in Tango.

`dateFormat`
`timeFormat`
`timestampFormat`

See also *MTCV*, page 171.

These options allow you to specify the formats for displaying and entering date, time, and timestamp values. The formats determine the default display formats of retrieved database values as well as those returned by meta tags. You can select a value from the drop-down menu.

## Debugging and Logging

The options in this screen set the defaults for debugging and logging Tango application files.

`debugMode`

See also *MTCV*, page 173.

If this option is set to the default, it allows the debug mode to be set within an application file. You can also force debugging on or off for all application files.

`loggingLevel`

See also *MTCV*, page 178.

By default, logging is turned off, but can be adjusted for different options of logging.

`logDir`

This option sets the folder used for logging. The log folder should be unique for every Tango Server running on the same machine. The default value is `{default path}log.{name of server}`.

## Execution

The options in this screen set the defaults for executing Tango application files.

`queryTimeout`

See also *MTCV*, page 183.

This option causes application file executions that exceed the specified number of seconds to time out and return the HTML page

specified in `timeoutHTML`. You can select a value from the drop-down menu or enter a custom value. The default value is five minutes.

`maxActions`

See also *MTCV*, page 181.

If the option is set to a positive number (the default is zero), the number of Tango actions executed so far by a query is checked against the value of this variable. If the number of actions exceeds the value, the query aborts and returns an error. You can select a value from the drop-down menu or enter a custom value.

`returnDepth`

See also *MTCV*, page 184.

This option sets the maximum number of branch levels that you can have in a Tango application file. The default value is 20. This applies to Branch actions which have the Return option set. It specifies the number of returns that can be outstanding at any time. If this limit is exceeded during an application file execution, an error occurs. You can select a value from the drop-down menu or enter a custom value.

## Feature Switches

The options in this screen are special system configuration variables that enable or disable Tango features. Possible values for all these switches are enabled or disabled (`on` and `off`). All these switches are enabled by default.

`javaSwitch`

For more information, see “Configuring a Java Action” on page 269.

This option tells Tango whether or not to execute Java.

`javaScriptSwitch`

For more information, see “Executing JavaScript” on page 264.

This option tells Tango whether or not to execute JavaScript in Tango Server (that is, JavaScript that is delineated with the `<@SCRIPT>` tag or using the Script action).

**Note** This is different from JavaScript that is passed on to and executed by the browser using the `<SCRIPT>` HTML tag.

`fileReadSwitch`

For more information, see “Setting Up Read Options” on page 281.

This option tells Tango whether or not to allow Tango to read external files using the File action or the `<@INCLUDE>` meta tag.

`fileWriteSwitch`

For more information, see “Setting Up Write Options” on page 282.

This option tells Tango whether or not to allow Tango to write to external files using the File action.

`fileDeleteSwitch`

For more information, see “Setting Up Delete Options” on page 283.

This option tells Tango whether or not to allow Tango to delete external files using the File action.

`externalSwitch`

For more information, see “Extending Tango Functionality” on page 263.

This option tells Tango whether or not to allow External actions in Tango.

`docsSwitch`

See also *MTCV*, page 73.

This option tells Tango whether or not to allow the use of the `<@DOCS>` meta tag, which returns the contents of an application file. A switch is provided because examining the contents of any application file could potentially be a security issue.

`mailSwitch`

For more information, see “Sending Electronic Mail From Tango” on page 275.

This option tells Tango whether or not to allows e-mail messages to be generated within Tango using the Mail action.

`passThroughSwitch`

For more information, see “Connecting to Data Sources” on page 93.

This option controls whether the user can use meta tags (such as `<@POSTARG>`) in the user and password fields when connecting to a data source using an application file. If this switch is disabled, the user name and password must be hard coded. This provides a certain amount of security for databases accessed through Tango.

## File Paths

The options in this screen set the default paths to certain files used by Tango Server.

`defaultErrorFile`

See also *MTCV*, page 175.

If this file exists, Tango uses the contents of this file as the error message returned to a user whenever an error condition occurs within an application file (unless you have specified Error HTML within the application file itself). You can edit the file with a text or HTML editor. The default value of this configuration variable is `{default path}error.htx`.

`timeoutHTML`

See also *MTCV*, page 188.

This is the HTML document to return to the user if an application file execution times out. You can edit the `timeout.html` document with a text or HTML editor if you want to change the HTML that is returned to the user. The default value of this configuration variable is `{default path}timeout.html`

`headerFile`

See also *MTCV*, page 177.

Tango uses the contents of this file as the HTTP header which is returned every time a request is sent to the Web browser. This is either an HTTP header targeted to the Web server under normal operation, or an HTTP header tailored for the client browser in the case of non-parsed headers being used. The default value of this configuration variable is `{default path}header.htx`.

`logDir`

This variable is also specified in the Debugging and Logging screen. See “logDir” on page 320.

`varCachePath`

See also *MTCV*, page 190.

This option specifies a folder to which Tango writes all variables when it is shutdown, and re-reads those variables from when Tango is started. The default value of this configuration variable is `{default path}variables.{name of server}`.

See also *MTCV*, page 191.

**Note** Variables continue to expire in the usual fashion; if you restart Tango after the specified user timeout period has elapsed, all the user variables immediately expire upon being reloaded.

`crontabFile`

For more information, see “Timed URL Processing With Tango Server” on page 312.

The cron, or Timed URL processing mechanism allows you to execute specified URLs at specified times. Only HTTP-type URLs are supported. The URLs may point to the local HTTP and Tango server or a remote one. The result HTML of the executed URL is discarded.

For more information, see “Format of the crontab File” on page 313.

The specifications are stored the file specified by this configuration file setting, conventionally called `crontab`, which is read on Tango Server startup. The cron service becomes active once per minute, and executes, one by one, all the URLs that satisfy the specification conditions between the previous and present activation.

## HTTP

The option in this screen sets the format of the default HTTP header.

`headerFile`

This variable is also found in the File Paths screen. See “headerFile” on page 323.

`userAgent`

See also *MTCV*, page 188.

This option changes the value of the user-agent value in HTTP requests.

## Mail

The options in this screen set the e-mail defaults for Tango Server.

`mailServer`

See also *MTCV*, page 180. This option in the configuration file sets the e-mail server to use with the Mail action of Tango Server. The default value is `localhost`.

`mailPort`

See also *MTCV*, page 180. This option in the configuration file sets the port for the e-mail server used with the Mail action. The default value is `25`.

`mailDefaultFrom`

See also *MTCV*, page 180. This option in the configuration file sets the default "From" e-mail address for any e-mail generated with Tango Server. The default value is set when Tango is installed under Mac OS.

## Memory

The options in this screen set the caching defaults for Tango Server.

`cache`

See also *MTCV*, page 167. This option enables or disables caching of application files and `<@INCLUDE>` files. The default value is enabled (`true`).

`cacheSize`

See also *MTCV*, page 167. This option in the configuration file sets the cache size for files in bytes. You can select from the drop-down menu or enter a custom value. The default value is 2,000,000 bytes.

## Miscellaneous

The options in this screen set certain options that are not specified elsewhere.

`encodeResults`

See also *MTCV*, page 176. This option tells Tango whether or not to encode the output sent to the Web browser by Tango by changing all high-bit characters to their encoded forms. For example, "é" is encoded in HTML as `&#233;`. If you would like to send binary data, or are using a character set other than ISO Latin-1, you can disable this option. The default value is enabled (`true`).

`threadPoolSize`

See also *MTCV*, page 186. This option determines the number of worker threads that the Tango Server allocates to process requests. The default value of this variable is `20`.

## Number Formats

The options in this screen set various defaults for how numbers are used within Tango. Some of these options are

`decimalChar`

See also *MTCV*, page 174.

This option sets the decimal point character. You can select a value from the drop-down menu or enter a custom value. The default value is ".".

`thousandsChar`

See also *MTCV*, page 186.

This option sets the separator character for numerals greater than one thousand. You can select a value from the drop-down menu or enter a custom value. The default value is ",".

`currencyChar`

See also *MTCV*, page 169.

This option sets the character used when numbers are specified as currency. You can select a value from the drop-down menu or enter a custom value. The default value is "$".

`staticNumericChars`

See the *Meta Tags and Configuration Variables Addendum*.

This option sets whether to not allow changes to the numeric format configuration variables during execution of an application file. The default value is true (checked), which gives better server performance.

## Server Start and Stop

The options in this screen set the defaults for the startup and shutdown URL mechanism, which initializes and finalizes the Tango Server environment.

`startupURL`
`shutdownURL`
`startStopTimeout`

For more information, see "Startup and Shutdown URL Processing" on page 314.

`startupUrl` specifies the URL which is executed on Tango Server startup. `shutdownUrl` specifies the URL which is executed upon Tango Server shutdown request.

`startstopTimeout` specifies, in seconds, the time which Tango Server is allowed to wait for Startup / Shutdown URL to complete. By default, the waiting period is one minute. Select another value from the drop-down menu or enter a custom value.

## TCP/IP

The options in this screen set the defaults for TCP / IP networking on Windows and Unix systems.

`validHosts`

See also *MTCV*, page 190.

This configuration file option sets a list of hosts for which Tango Server accepts Tango CGI connections. The names of the hosts go

here, separated by a colon (":"). Setting a list of valid hosts prevents an arbitrary user on your network or the Internet from using your Tango Server. The default value is `localhost`.

`listenerPort`

See also *MTCV*, page 178. This is the port number used by Tango Server to listen for requests from the Tango CGI. This number can be any valid port number that is not currently in use on your system. (Various UNIX operating systems and applications reserve ports.) The default value is `18000`.

## Variables

The options in this screen set the defaults for variables and variable keying in Tango.

`userKey`, `altUserKey`

See also *MTCV*, page 189. This option tells Tango Server what piece of information to use to identify a user when assigning to and evaluating user variables. The value of the configuration variable `userKey` is the default key for user variables. If its contents evaluate to empty, `altUserKey` is used instead.

The default value of `userKey` is `<@USERREFERENCE>`, indicating that user variables are keyed on the Tango user reference ID assigned to each user. The default value of `altUserKey`, `<@CGIPARAM CLIENT_IP>`, distinguishes users by IP address.

`domainScopeKey`

See also *MTCV*, page 175. This option sets the key for the domain variable scope. The default value, `<@CIPHER action=hash str=<@CGIPARAM server_name>>`, sets an encrypted key for the domain based on the name of the server.

`variableTimeout`

See also *MTCV*, page 191. This option sets the number of minutes before user variables expire. You can select a value from the drop-down menu or enter a custom value. The default value is 30 minutes.

`defaultScope`

See also *MTCV*, page 175. This option indicates the scope that is used when an assignment is made to a non-existent variable and no scope has been specified. The default value is `user`.

## Custom

This screen allows you to access all of the Tango Server system configuration variables by name and with their values specified fully, that is, not by drop-down menu shortcuts. For more details about the possible values that these configuration variables can

have, see Chapter 2, “Configuration Variables,” in the *Meta Tags and Configuration Variables* manual.

In addition to all the variables that are listed previously, the following variables can be set:

`license`

This option ensures the product is licensed and that the server only runs if a valid license number is entered.

`logToResults`

See also *MTCV*, page 179.

If this option is set to true, log information (level set by the configuration variable `loggingLevel` with different scopes) is returned at the bottom of each hit, rather than being written to the log file specified in the configuration variable `logDir`. The default value is `false`.

`persistentRestart`

See also *MTCV*, page 182.

This option controls how the server handles an automatic restart. An automatic restart is initiated when `maxHeapSize` is exceeded and when Tango detects a problem with servicing requests.

When set to `true`, the server first attempts to completely shut down the running server before restarting a new one. All variables in use at the time of the shutdown are preserved. This is the default value.

When set to `false`, a new server is started immediately, even before the old one is stopped. This setting ensures high server availability, but variables from the old server instance are not available in the new one.

`postargFilter`

See also *MTCV*, page 183.

This option sets the character(s) stripped out of all data into Tango through postargs.

`FMDatabaseDir`

See also *MTCV*, page 177.

*Mac OS*: This option is a path telling Tango where to look for local FileMaker Pro databases. When Tango tries to connect to a local data source and finds that the database is not open, it looks in this folder and opens the database (if present).

`maxSessions`

See also *MTCV*, page 181.

*Mac OS only*: This option determines the maximum number of sessions Tango Server opens for DAM data sources on the Mac OS. It accepts any positive integer as a value. A value of 0 (the default) indicates no maximum.

`enableTangoUserDocs`

*UNIX only*: This option in the configuration file sets whether user-owned documents work. In secure environments, you may only want administrative Tango application files to be executed. Possible values are `true` and `false`. The default value is `false`.

An example of a user-owned document is:

```
http//www.isp.com/cgi_bin/Tango.cgi/~user/some_taf
```

**Note** The "~" is often configurable by Web servers. Tango transparently follows the path that the Web server uses in place of the "~".

`maxHeapSize`

See also *MTCV*, page 181.

*UNIX only*: Tango Server restarts itself in a clean state if its heap size exceeds this specified limit. The default value is `2000000` bytes.

# Editing the Tango Server Configuration File Directly

The Tango Server configuration file (`Tango 3 Server Preferences`) contains the defaults for many Tango Server options. These options are set by default. You generally do not need to edit this file directly.

All of the configuration file settings can be set using Tango system configuration variables. The format of configuration file options and configuration variables is almost identical, except that configuration file options are always specified in uppercase, whereas configuration variables are generally specified in mixed case.

On Windows, this file is called `t3server.ini`, and is by default located in `C:\Windows\Tango3`, or `C:\Winnt\Tango3\`. (On UNIX, the file is by default located in `/var/opt/EDI/`. On Mac OS, the file is called `Tango 3 Server Preferences` and is located in the `Preferences` folder, inside the `System Folder` on the startup hard disk. ).

If necessary, you can edit this file directly with a text editor. Tango Server does not read this file until it starts up.

**Caution** Tango Server overwrites this configuration file when it quits if any configuration variables have changed. Therefore, you should not edit this file while Tango Server is running; Tango may overwrite the configuration file that you have changed, destroying your changes.

Always quit Tango Server before editing the configuration file directly.

The text file contains lines with the following structure and default values:

```
[Tango Definitions]
Tango3 Server=Default Tango Server
Tango3.acgi=

[Tango3 Server]
...configuration file line entries...
```

These lines are at the top of the configuration file and tell Tango Server that the current file sets the defaults for one or more Tango Servers on the same machine.

Under the `[Tango Definitions]` heading is a list of the different Tango Servers (if there are more than one) followed by an equals sign and an optional comment. The `[Tango3 Server]` block which follows sets the defaults for that copy of Tango Server.

Additional configuration line entries for different copies of Tango Server would be in the same file, preceded by a different name in square brackets, which matches the entry in the definitions section (for example, `[Tango3 Server2]`).

The only configuration variable that you may need to edit directly is `configPassword`.

```
configPassword
```

See also *MTCV*, page 168. The configuration variable, `configPassword` with user scope and `configPassword` with system scope must match in order for the user to modify system configuration variables, for example, in the `config.taf` application file.

# Tango CGI or Plug-in Configuration File

The `t3client.ini` file, which resides in `C:\Winnt\Tango3\` , `C:\Windows\Tango3\`, or`/var/opt/EDI/`, controls the configuration of the Tango CGI program or Tango Web server plug-in; that is, `t3cgi.exe` (Windows NT), `t3.cgi` (UNIX), `t3ns.dll` (Netscape Server plug-in) or `t3iis.dll` (Microsoft Internet Information Server plug-in). It is used to define what Tango Server process the program can talk to.

The structure of this file is similar to the Tango Server configuration file.

```
[Tango CGI Definitions]
t3cgi.exe=Default CGI under Windows NT
...additional CGI entries...

[t3cgi.exe]
...configuration file line entries...

[additional CGI name]
...configuration file line entries...
```

These lines are at the top of the configuration file and tell the Tango that the current file sets the defaults for one or more Tango CGI programs on the same machine.

Under the `[Tango CGI Definitions]` heading is a list of the different Tango CGI programs (if there are more than one) followed by an equals sign and an optional comment. The `[t3cgi.exe]` block which follows sets the defaults for that CGI.

Additional configuration line entries for different CGI programs would be in the same file, preceded by a different name in square brackets, which matches the entry in the definitions section (for example, `[t3cgialt.exe]`).

If you are using the plug-ins and not the CGI program, use `t3ns.dll` or `t3iis.dll` here. If you are running Tango Server under UNIX, use `t3.cgi` here.

The following describes the configuration file line entries.

```
TANGO_HOST=localhost
```

This line in the configuration file specifies the name of the host that Tango CGI connects to. This is the machine name or IP address where the Tango Server process is running.

```
TANGO_PORT=18000:18001:18002
```

For more information, see “listenerPort” on page 326.

This is the port number to which Tango CGI will try to connect. This should correspond to the value in `cgiListenerPort` in the Tango Server configuration file. This can be a list of ports, delineated with the colon (“:”) character, which would specify multiple Tango Servers. Tango CGI randomly selects one of the ports to connect to. If the connection fails, it continues to try until the remaining ports until all have been exhausted. This can be used to scale a site that *does not* require variable sharing among multiple servers.

APPENDIX A

# *Glossary of Terms*

A

*An Alphabetical Reference of Common Tango and Internet Terms*

**action**

This is short for *Tango action*. Tango Editor has a suite of actions which do many different tasks, including: getting data from or sending data to a database, invoking external actions (such as reading and writing files and sending email), and controlling application file execution. Actions form the basis of an application file in Tango Editor.

**Apple Event**

*Mac OS only*: An Apple Event is a high-level event that conforms to the Apple Event Interprocess Messaging Protocol. The Apple Event Manager uses the services of the Event Manager to send Apple events between applications on the same computer, between applications on remote computers, or from an application to itself. For more information, visit:

```
http://www.apple.com/
```

**application file**

Application files are written using Tango Editor and are composed of one or a series of actions that are executed by Tango Server. Each action's reaction or response from a database, server, external program, etc. can be posted in HTML. When an application file is completed, the results are returned to the client.

**applet**

A small Java program that can be embedded in an HTML page. Applets differ from full-fledged Java applications in that they are not allowed to access certain resources on the local computer, such as files and serial devices, and are prohibited from communicating with most other computers across a network. The current rule is that an applet can only make an Internet connection to the computer from which the applet was sent.

**ASCII**

**American Standard Code for Information Interchange**
This is the *de facto* world-wide standard for the code numbers used by computers to represent all the upper- and lower-case Latin letters, numbers, punctuation, and related data. There are 128 standard ASCII codes each of which can be represented by a seven-digit binary number.

**bandwidth**

This is the measure of how much you can send through a connection, usually measured in bits per second. A full page of English text is about 16,000 bits. A moderately fast modem can move about 15,000 bits in one second. Full-motion full-screen video would require roughly 10,000,000 bits per second, depending on compression.

**baud**

In common usage the baud rate of a modem is how many bits it can send or receive per second. Technically, baud is the number of times per second that the carrier signal shifts value. For example, a 1200 bit-per-second modem actually runs at 300 baud, but it moves 4 bits per baud (4 x 300 = 1200 bits per second).

**bit**

**binary digit**
A single digit number in base-2, in other words, either a one or a zero. The smallest unit of computerized data.

**bps**

**bits per second**
A measurement of how fast data is moved from one place to another. A 28.8K modem can move 28,800 bits per second.

**CGI**

**Common Gateway Interface**
A set of rules that describes how a Web server communicates with another piece of software, often on the same machine, and how the other piece of software (the "CGI program") talks to the Web server. Any piece of software can be a CGI program if it handles input and output according to the CGI standard.

Usually a CGI program is a small program that takes data from a Web server and does something with it, like putting the content of a form into an e-mail message, or turning the data into a database query.

You can often see that a CGI program is being used when "cgi-bin" appears in a URL.

**cgi-bin**

The most common name of a directory on a Web server in which CGI programs are stored.

The "bin" part of "cgi-bin" is a shorthand for "binary", because once upon a time, most programs were referred to as "binaries". In real life, most programs found in cgi-bin directories are text files—scripts that are executed by binaries located elsewhere on the same machine.

**client**

A software program that is used to contact and get data from a server software program on another computer, often across a network or the Internet. Each client program is designed to work with one or more specific kinds of server programs, and each server requires a specific kind of client. A Web browser is a specific kind of client.

**configuration variables**

Special values that control aspects of Tango behavior. System variables affect system wide settings. They apply to all users of Tango Server.

**cookie**

The most common meaning of *cookie* on the Internet refers to a piece of information sent by a Web server to a Web browser that the browser software is expected to save and to send back to the server whenever the browser makes additional requests from the server.

Depending on the type of cookie used and the browser's settings, the browser may accept or not accept the cookie, and may save the cookie for either a short time or a long time.

Cookies might contain login or registration information, on-line shopping cart information, or user preferences.

When a server receives a request from a browser that includes a cookie, the server is able to use the information stored in the cookie. For example, the server might customize what is sent back to the user, or keep a log of particular users' requests.

Cookies are usually set to expire after a predetermined amount of time and are usually saved in memory until the browser software is closed down, at which time they may be saved to disk if their expire time has not been reached.

**data source**

An abstraction or description of the database that Tango Editor and Tango Server are referencing.

**domain name**

The unique name that identifies an Internet site. Domain names always have two or more parts separated by dots. The part on the left is the most specific, and the part on the right is the most general. A given machine may have more than one domain name, but a given domain name points to only one machine. For example, the domain names

```
example.com
training.example.com
mail.example.com
```

can all refer to the same machine, but each domain name can refer to no more than one machine.

Usually, all of the machines on a given network have the same thing as the right-hand portion of their domain names (`example.com` in the examples above). It is also possible for a domain name to exist but not be connected to an actual machine. This is often done so that a group or business can have an Internet e-mail address without having to establish a real Internet site. In these cases, some real Internet machine must handle the mail on behalf of the listed domain name.

**Domain Name Server (DNS)**

A machine on the Internet that converts ("resolves") domain names to IP address numbers.

**e-mail**

Messages, usually text, sent from one person to another via computer. The "e" is for "electronic".

**firewall**

A combination of hardware and software that separates a LAN into two or more parts for security purposes.

**gateway**

A hardware or software setup that translates between two dissimilar protocols. For example, Prodigy has a gateway that translates between its internal, proprietary e-mail format and Internet e-mail format. Another meaning of gateway is to describe any mechanism for providing access to another system, for example, AOL might be called a gateway to the Internet.

**hit**

A single request from a Web browser for a single item from a Web server; thus for a Web browser to display a page that contains three graphics, four hits would occur at the server—one for the HTML page, and one for each of the three graphics.

Hits are often used as a rough measure of load on a server, such as the server gets 300,000 hits per month. Because each hit can represent anything from a request for a tiny document (or even a request for a missing document) all the way to a request that requires some significant extra processing (such as a complex search request), the actual load on a machine from one hit is almost impossible to define.

**home page**

The Web page that your browser is set to use when it starts up, or the main Web page for a business, organization, person, or simply the main page out of a collection of Web pages.

**host**

Any computer on a network that is a repository for services available to other computers on the network. It is quite common to have one host machine provide several services, such as WWW and Usenet (Netnews).

**HTML**

**HyperText Markup Language**
The coding language used to create hypertext documents for use on the World Wide Web. HTML looks a lot like old-fashioned typesetting code, where you surround a block of text with codes that indicate how it should appear. Additionally, in HTML you can specify that a block of text, or a word, is linked to another file on the Internet. HTML files are meant to be viewed using a World Wide Web client program, such as Netscape Navigator or Microsoft Internet Explorer.

**HTTP**

**HyperText Transfer Protocol**
The protocol for moving hypertext files across the Internet. Requires a HTTP client program on one end, and an HTTP server program on the other end. HTTP is the most important protocol used in the World Wide Web.

**hypertext**

Generally, any text that contains links to other documents—words or phrases in the document that can be chosen by a reader and that cause another document to be retrieved and displayed.

**Internet**

The vast collection of interconnected networks that all use the TCP/IP protocols and that evolved from the ARPANET of the late sixties and early seventies.

**intranet**

A private network inside a company or organization that uses the same kinds of software that you would find on the public Internet, but that is only for internal use.

As the Internet has become more popular many of the tools used on the Internet are being used in private networks, for example, many companies have Web servers that are available only to employees.

**ISDN**

**Integrated Services Digital Network**
A way to move more data over existing regular phone lines. It can provide speeds of roughly 128,000 bits per second over regular phone lines. In practice, most people will be limited to 56,000 or 64,000 bits per second.

**IP number**

**Internet Protocol Number**
A unique number consisting of four parts separated by dots, such as `207.107.95.106`. Every machine that is on the Internet has a unique IP number—if a machine does not have an IP number, it is not really on the Internet. Most machines also have one or more domain names that are easier for people to remember.

**Java**

Java is a network-oriented programming language invented by Sun Microsystems that is specifically designed for writing programs that can be safely downloaded to your computer through the Internet and immediately run without fear of viruses or other harm to your computer or files. Using small Java programs (called *applets*), Web pages can include functions such as animations, calculators, and other fancy tricks.

**Java bean**

A component technology for Java that lets developers create reusable software objects. These objects can be shared. A database vendor can create a Java bean to support its software, and other developers can easily drop the bean into their own projects.

**Java class**

In Java, a type that defines the implementation of a particular kind of object. A class definition defines instance and class variables and methods, as well as specifying the interfaces the class implements and the immediate superclass of the class. If the superclass is not explicitly specified, the superclass will implicitly be `Object`.

**LAN**

**Local Area Network**
A computer network limited to the immediate area, usually the same building or floor of a building.

**meta tag**

The basic component of a tag language unique to Tango Server. Meta tags communicate with Tango Server in the same way that HTML communicates with a Web server.

**MIME**

**Multipurpose Internet Mail Extensions**
A standard for attaching non-text files to standard Internet mail messages. Non-text files include graphics, spreadsheets, formatted word-processor documents, sound files, and most other files.

An e-mail program is said to be MIME Compliant if it can both send and receive files using the MIME standard.

When non-text files are sent using the MIME standard they are converted (encoded) into text—although the resulting text is not readable.

Generally speaking, the MIME standard is a way of specifying both the type of file being sent (for example, a Quicktime video file), and the method that should be used to turn it back into its original form.

Besides e-mail software, the MIME standard is also universally used by Web Servers to identify the files they are sending to Web clients. In this way, new file formats can be accommodated simply by updating the browsers' list of pairs of MIME-types and appropriate software for handling each type.

**network**

Any time you connect two or more computers together so that they can share resources, you have a computer network. Connect two or more networks together and you have an internet.

**NNTP**

**Network News Transport Protocol**
The protocol used by client and server software to carry Usenet postings back and forth over a TCP/IP network. If you are using any of the more common software such as Netscape Navigator, Nuntius, Internet Explorer, or others to participate in newsgroups then you are benefiting from an NNTP connection.

**ODBC**

**Open Database Connectivity**
A standard set by Microsoft that allows applications to

communicate with a variety of databases from different vendors. An ODBC client application talks to the ODBC driver manager, which in turn talks to a database driver for a specific type of database.

**plug-in**

A (usually small) piece of software that adds features to a larger piece of software. A common example of a plug-in is for the Netscape Navigator browser and Web server. The idea behind plug-ins is that a small piece of software is loaded into memory by the larger program, adding a new feature, and that users need only install the few plug-ins that they need, out of a much larger pool of possibilities. Plug-ins are usually created by people other than the publishers of the software the plug-in works with.

**PPP**

**Point to Point Protocol**
Most well known as a protocol that allows a computer to use a regular telephone line and a modem to make TCP/IP connections and connect to the Internet.

**security certificate**

Information (often stored as a text file) that is used by the SSL protocol to establish a secure connection.

Security certificates contain information about who it belongs to, who it was issued by, a unique serial number or other unique identification, valid dates, and an encrypted fingerprint that can be used to verify the contents of the certificate.

For an SSL connection to be created, both sides must have a valid Security Certificate.

**scope**

In Tango, refers to an characteristic of variables that determines where they are valid: in local, user, cookie, domain, or system scope. See Chapter 7 for details.

In Java, a characteristic of an identifier that determines where the identifier can be used. Most identifiers in Java have either class or local scope. Instance and class variables and methods have class scope; they can be used outside the class and its subclasses only by prefixing them with an instance of the class or (for class variables and methods) with the class name. All other variables are declared within methods and have local scope; they can be used only within the enclosing block.

**server**

A computer, or a software package, that provides a specific kind of service to client software running on other computers. The term can refer to a particular piece of software, such as a Web server, or to the machine on which the software is running. A single server machine could have several different server software packages running on it, thus providing many different servers to clients on the network.

**SMTP**

**Simple Mail Transport Protocol**
The main protocol used to send mail on the Internet. SMTP consists of a set of rules for how a program sending mail and a program receiving mail should interact.

Almost all Internet mail is sent and received by clients and servers using SMTP, thus if one wanted to set up an e-mail server on the Internet one would look for e-mail server software that supports SMTP.

**SQL**

**Structured Query Language**
A specialized programming language for sending queries to databases. Most industrial-strength relational databases and many smaller database applications are addressed using SQL. Each specific application will have its own version of SQL implementing features unique to that application, but all SQL-capable databases support a common subset of SQL.

**SSL**

**Secure Sockets Layer**
A protocol designed by Netscape to enable encrypted, authenticated communications across the Internet.

SSL is used mostly (but not exclusively) in communications between Web browsers and Web servers. URLs that begin with “https” indicate that an SSL connection will be used.

SSL provides three important things: privacy, authentication, and message integrity.

In an SSL connection each side of the connection must have a Security Certificate, which each side’s software sends to the other. Each side then encrypts what it sends using information from both its own and the other side’s Certificate, ensuring that only the intended recipient can decrypt it, and that the other side can be sure the data came from the place it claims to have come from, and that the message has not been tampered with.

**Tango CGI**

The CGI that links Tango Server and your Web server. Not necessary if you use one of the Web server plug-ins.

**Tango Server**

Executes and serves up the application files by which clients can interact with HTML pages to perform a variety of tasks; for example, querying the data source.

**TCP/IP**

**Transmission Control Protocol/Internet Protocol**
This is the suite of protocols that defines the Internet. Originally designed for the UNIX operating system, TCP/IP software is now available for every major kind of computer operating system. To be truly on the Internet, your computer must have TCP/IP software.

**UNIX**

A computer operating system (the basic software running on a computer, underneath things like word processors and spreadsheets). UNIX is designed to be used by many people at the same time (it is multi-user) and has TCP/IP built-in. It is the most common operating system for servers on the Internet.

**URL**

**Uniform Resource Locator**
A standard way to give the address of any resource on the Internet, most commonly used for the World Wide Web (WWW). Here is an example of a URL:

```
http://www.example.com/
```

The most common way to use a HTTP-type URL is to enter it into a Web browser program, such as Netscape Navigator or Microsoft Internet Explorer.

**WWW**

**World Wide Web**
The universe of hypertext servers (HTTP servers) on the Internet, which are the servers that send encoded markup text (HTML) pages when requested by client Web browsers. HTML pages often link to graphics, sound files, and many other different kinds of file formats.

APPENDIX B

# *Using DLLs With Tango*

B

*Programmer Reference for Extending the Functionality of Tango Using DLLs*

This appendix provides information on creating Dynamic Link Libraries (DLLs) for use with the External action when executing Tango application files on the Windows platform. This information is provided for those programmers who want to extend the functionality of Tango Server through the use of DLLs.

## TExtParamBlock

Tango passes each function the following parameter block:

```
struct TExtParamBlock{
       DWORD ThreadId;
       DWORD CurrRow;
       DWORD CurrColumn;
       VOID *UserData;
       VOID *Reserved;
};
```

Tango uses this "extension parameter block" to communicate the current state to the DLL. The DLL uses it to track user data between invocations of the DLL. The members of the parameter block are:

- `DWORD ThreadId;`

  The ID of the calling thread allocated by Tango. The value is set by Tango; it may not be changed by the DLL.

- `DWORD CurrRow;`

  The row number currently processed by Tango. This value is incremented by Tango as it iterates through each row of the result set.

- `DWORD CurrColumn;`

  The column number currently processed by Tango. This value is incremented by Tango as it iterates through each column of a particular row of the result set.

- `VOID *UserData;`

  Contains any user defined data. If the DLL requires some memory on a per query basis, use the UserData member to keep a reference to the memory block. UserData is usually assigned at the time of `ExtSrcConnect` call. It must be freed in the `ExtSrcDisconnect` function.

- `VOID *Reserved;`

  Reserved for future use by Tango. Do not reference or set this member.

# DLL Functions

DLL writers must implement five functions in their DLL. A sixth function, used to process errors, is optional. Tango calls these functions to process specific events.

The prototypes for these functions are defined in the `ExtSrc.h` file, included with Tango.

These DLL functions are:

- `extern "C" _export int ExtSrcConnect(TExtParamBlock *param_block);`

  This function is called when the external data source is connected, usually the first time the External action that references the DLL is executed. The `UserData` member of the `TExtParamBlock` structure should be initialized at this point.

  This function must return one of the following values:

  `EXT_SRC_SUCCESS` (if connection is successful)

  `EXT_SRC_ERROR` (otherwise)

- `extern "C" _export int ExtSrcDisconnect(TExtParamBlock *param_block);`

  This function is called when the external data source is disconnected, usually when the Tango Service is stopped. This function provides the last opportunity to deallocate any memory referenced by the `UserData` member of the parameter block.

  This function must return one of the following values:

  `EXT_SRC_SUCCESS` (if disconnection is successful)

  `EXT_SRC_ERROR` (otherwise)

- `extern "C" _export int ExtSrcExecuteQuery(TExtParamBlock *param_block, char *p1, char *p2, char *p3);`

  The `ExtSrcExecuteQuery` function is called once for each time the External action is executed by Tango Server. This function either returns an error code or the number of columns in the result set arising from the execution of the DLL. The number of columns is used by the Tango Server to control when the `ExtSrcGetNextColumn` function is called.

For more information, see “Configuring a DLL Call” on page 266.

The declaration of this function depends on the number of parameters you intend to pass to the DLL. After the `param_block` parameter you need to include a char * parameter for each parameter defined in the External action window in Tango Editor. For example, the prototype shown above is for a DLL that has three parameters defined for it in the External action.

This function must return one of the following values:

`EXT_SRC_ERROR` (in case of error)

result set quantity (zero or greater number identifying the number of columns in the result set)

- `extern "C" _export int ExtSrcFetchNextRow(TExtParamBlock *param_block);`

  This function is called by Tango Server once for each row of the result set created by the `ExtSrcExecuteQuery` function. The result of this function determines the number of times it is called: Tango Server continues to call `ExtSrcFetchNextRow` until the function returns `EXT_SRC_NODATA` or `EXT_SRC_ERROR`.

  This function does not return data to Tango Server. It should be used by the DLL to load or prepare the data for retrieval. After calling this function, the `ExtSrcGetNextColumn` function is called to retrieve the data from each column. `ExtSrcGetNextColumn` is called once for each column in the result set; the number of columns is determined by the result of the `ExtSrcExecuteQuery` function.

  This function must return one of the following values:

  `EXT_SRC_SUCCESS` (if the row is retrieved successfully)

  `EXT_SRC_NODATA` (if there are no rows remaining to return)

  `EXT_SRC_ERROR` (if an error occurs)

- `extern "C" _export int ExtSrcGetNextColumn(TExtParamBlock *param_block, UCHAR *buffer, DWORD blen, DWORD *actlen);`

  This function is called by Tango Server once for each column of the result set for each row fetched by the `ExtSrcFetchNextRow` function.The number of columns is determined by the result of the `ExtSrcExecuteQuery` function.

This function has the following parameters:

`TExtParamBlock *param_block` (pointer to the external action parameter block)

`UCHAR *buffer` (pointer to a 32K buffer allocated by Tango)

`DWORD blen` (size of the buffer allocated by Tango (currently set to 32K))

`DWORD *actlen` (actual size of the data written by the DLL to the buffer)

This function must return one of the following values:

`EXT_SRC_SUCCESS` (if the column's data is retrieved successfully)

`EXT_SRC_ERROR` (if an error occurs)

- `extern "C" _export int ExtSrcErrorCode(TExtParamBlock *param_block);`

  This is an optional function. If implemented, Tango calls `ExtSrcErrorCode` whenever one of the other DLL functions returns `EXT_SRC_ERROR` to Tango.

APPENDIX C

# *Using Java With Tango*

C

*Java Actions and the Java Action Server*

This appendix provides additional information on calling Java classes from Tango for use with the External action in Tango. This information is provided for those programmers who want to extend the functionality of Tango through the use of Java.

# Installing Java Action Server

To execute Java actions from Tango, you need to install and run the Java Action Server (JAS). The JAS is a Java application that accepts requests from Tango to execute Java class files, and returns the results of that execution back to Tango. The JAS can run on the same machine as Tango, or it can run on another, network-accessible (via TCP/IP) machine.

**Note** JAS requires that you have installed a Java virtual machine (VM), such as the one that comes with the Java Development Kit (JDK) 1.1 from Sun Microsystems (`http://java.sun.com/`).

When you install Tango Server, the following directory hierarchy is present in your destination folder:

```
java\com\everyware\tango\jas
```

Refer to your Java documentation for details on modifying your class path.

These directories have all of the files necessary to run the JAS. In order for the Java VM to find these files, you need to add this hierarchy to your Java class path. Using the Sun JDK on Windows NT, for example, you would:

1. Open the System control panel.
2. Click the Environment tab.
3. Locate (or add) a variable called `CLASSPATH`.
4. If your Tango installation is in `Tango 3.0` (for example), add the following to `CLASSPATH`:

   ```
   C:\Tango 3.0\java
   ```

5. Apply your changes and close the control panel.

The Java server is run from the command line. Open a new DOS window (or shell on UNIX), and set the current directory to the `jas` directory. Then execute the following command:

```
java com.everyware.tango.jas.JASMain
```

On UNIX, you probably want to append an ampersand to cause the server to run in the background. You should now be able to use Tango to connect to JAS and execute Java class files. You can test your setup by running the `JAStest.taf` application file installed with Tango Server (`/JavaDemo/JAStest.taf`).

## Configuring JAS

There are several parameters that can be configured in the Java server to tune its performance. These parameters are passed on the command line, or provided in the `JAS.properties` file that resides in the same directory as the `JASMain` class file. All the parameters can be found by passing `-help` to `JASMain`, but the important ones are:

- `-port p`

  Listens on port p for requests from Tango (default is 4000).

- `-serversockettimeout s`

  Specifies the maximum number of seconds to wait for a request, after which JAS quits.

- `-actiontimeout s`

  Specifies the maximum number of seconds to wait for a class file to execute.

- `-trace`

  Logs detailed information on server activity for debugging.

# Creating Java Action Classes

A Java class or JavaBean that is specified in a Java action must extend the class `com.everyware.tango.jas.Action` and provide an implementation for the `customProcessing` method:

```
public class Foo extends Action {
       // provide an implementation of customProcessing
       //to do the work
       public void customProcessing(String argv[]) {
        // do your stuff here...
       }
}
```

The single `String` array parameter is the list of parameters that are specified in the Java action in the Tango application file. The Action class provides several methods that you would use in your action processing to provide result data to Tango. These methods are:

- `protected void setNumColumns(int)`

  Indicate the number of columns to be output in the class results.

- `protected void newRow()`

  Start a new row in the output.

- `protected void newColumn(String)`

  Start a new column in the current row containing the given String.

There are two example class files in the `jas` directory, `Hello.java` and `Echo.java`, that illustrate using these methods. The `JAStest.taf` application file demonstrates using these classes in Tango.

**Note** When specifying the path to your class file in the Java action, you must use the Java "dot" notation (for example, `com.everyware.tango.jas.Hello`), and the path must be accessible from `CLASSPATH`.

JavaBeans are loaded from their serialized form, so not only must you provide a class file but you must also have a `.ser` file for that class.

APPENDIX D

# Tango Server Error Codes

D

*A Listing of Tango Server Error Numbers and Messages*

During execution of an application file by Tango Server, you may encounter certain errors conditions. This appendix lists the main error numbers and messages generated by Tango Server for the various error conditions.

**Note** The error numbers apply only if the type of error is *internal*. For other types of errors (for example, DBMS), consult the appropriate documentation (for example, database driver or server).

| Main Error Number | Message |
|---|---|
| **-1** | There was not enough memory to complete the requested operation. |
| **-2** | The specified object was not found. |
| **-3** | The application file was either missing or invalid. |
| **-4** | Unable to connect to the specified data source.Verify that data source is properly configured and that database server is online. |
| **-5** | Bad application file format version. |
| **-8** | Invalid value specified. Previous value has been used. |
| **-9** | Invalid or empty variable key. |
| **-10** | Invalid or empty variable name. |
| **-11** | Invalid or empty array name. |
| **-12** | Missing or invalid rows loop. |
| **-13** | Cannot initialize the JavaScript runtime. |
| **-14** | Invalid scope for this script. |
| **-15** | Script execution timeout. |
| **-16** | Begin Transaction encountered while existing transaction still open. |
| **-17** | End Transaction encountered with no open transaction. |
| **-18** | Error during expression evaluation. |
| **-19** | The maximum number of concurrent requests has been exceeded. |
| **-20** | The application file tag nesting exceeded limit. |
| **-21** | Loop execution timeout. |
| **-22** | The specified script language is not supported. |
| **-23** | Perl interpreter detected a syntax or runtime error. |
| **-101** | General error during data source operation. |
| **-102** | No data source connection exists. |
| **-103** | Connection to data source already exists. |
| **-104** | No data found. |
| **-105** | Invalid column number specified.Column number not in range of selected columns. |
| **-106** | Too many sessions for this server. Please try again later. |

| Main Error Number | Message |
|---|---|
| **-107** | Could not open specified database. Verify database name and ensure proper access privileges. |
| **-108** | Maximum licensed number of connections exceeded. Please try again later. |
| **-109** | This type of data source is not supported by the server license. |
| **-110** | The specified data source cannot be found. |
| **-111** | Invalid meta tag for this action. A data source is needed here. |
| **-112** | The data source operation took too long to execute. Try adjusting the server timeout parameter. |
| **-113** | Unable to communicate with the specified data source. The existing connection was lost. Please try again. |
| **-114** | The file specified is not part of the application designated by your server license. |
| **-116** | Could not write to configuration file. Check the file permissions. |
| **-117** | This action requires a data source. |
| **-201** | Failed to connect to gateway. |
| **-202** | There are no gateways currently defined. |
| **-203** | A gateway with that name already exists. |
| **-204** | Failed to remove gateway. |
| **-205** | A data source with that name already exists. |
| **-206** | There are no data sources currently defined. |
| **-301** | The specified file or directory does not exist. |
| **-302** | A permissions error occurred while trying to access the specified file. |
| **-303** | Can not open the specified file. |
| **-304** | Unable to obtain a write lock on the file. |
| **-305** | The specified file already exists. |
| **-306** | No file name is specified. |
| **-321** | Unable to connect to the specified SMTP server. |
| **-322** | Mail messages must have a From address. |
| **-323** | Tango supports only US-ASCII (7-bit) characters in message content. |
| **-324** | Connected to SMTP server but a communication error occurred. |

| Main Error Number | Message |
|---|---|
| **-325** | Mail messages must have a destination address. |
| **-401** | This feature has been disabled by the administrator. |
| **-501** | Only a branch to a top-level action is allowed when branching to different file. |
| **-502** | The number of nested returning Branch actions exceeds the limit. Check for an endless loop or increase the returnDepth configuration variable value. |
| **-503** | The file was not loaded because it has a corrupt structure. |
| **-504** | The application file specified in this Branch action cannot be found. |
| **-505** | The Branch destination action cannot be found in the specified application file. |
| **-506** | The number of actions executed so far exceeds the limit. Check for an endless loop or increase the maxActions configuration variable value. |
| **-510** | The structure of the application file is corrupt. |
| **-511** | This action may not have a child. |
| **-512** | This action has a malformed structure. |
| **-513** | The branch destination cannot be found or is at an invalid level. |
| **-514** | The Break action is valid only within For and While Loop actions and in groups. |
| **-515** | The Elseif/Else action is valid only when preceded by If action. |
| **-520** | A NULL node was encountered. |
| **-521** | An empty node was encountered. |
| **-522** | A container-type node expected but not found. |
| **-601** | System scope is for configuration variables only. Try using domain scope instead. |
| **-602** | The domain scope is not defined. Set the value of domainScopeKey. |
| **-603** | The array subscript is not within the range for the defined array. |
| **-604** | The string used to construct the array is invalid. Check the delimiters and make sure the dimensions match. |
| **-605** | You cannot apply array subscript operations on scalar variables. |
| **-606** | You do not have the correct password to set system variables. |
| **-607** | You cannot get the value of the configuration password. |

| Main Error Number | Message |
|---|---|
| **-608** | You cannot purge the contents of the system scope. |
| **-609** | You may not set configuration variables in the cookie scope. Try using the user or local scope instead. |
| **-610** | Destination's dimensions do not match the source's for array section assignment. |
| **-611** | Row and column dimensions within the <@ARRAY> tag must be greater than 0 if there is no initialization string. |
| **-612** | The initialization string does not match the specified row and column dimensions, or the row and column dimensions are inconsistent within the initialization string. |
| **-613** | The row and column delimiters must be fully unique if the array dimensions are not specified. |
| **-614** | An array was expected as a parameter. |
| **-615** | Parameter arrays must have the same number of columns. |
| **-616** | A scalar cannot be pushed into an array with multiple columns. |
| **-617** | A value to add must be specified. |
| **-618** | The COLS argument of an <@SORT> tag cannot be parsed. |
| **-700** | Invalid outer join. |
| **-701** | A table specified is an inner table to more than one outer table. |
| **-702** | Outer join specified creates cyclic join relationships. |

# Index

## A

## B



## C

## D

## E



## F

## G



## H



## I

## J



## K



## L



## M

## N

## O



## P

## Q



## R

## S

## T

## U