

KINTANA™

# Using Commands and Tokens

**Version 5.0.0**

Publication Number: KintanaCommands-0603A

Kintana, Inc. and all its licensors retain all ownership rights to the software programs and related documentation offered by Kintana. Use of Kintana's software is governed by the license agreement accompanying such Kintana software. The Kintana software code is a confidential trade secret of Kintana and you may not attempt to decipher or decompile Kintana software or knowingly allow others to do so. Information necessary to achieve the interoperability of the Kintana software with other programs may be obtained from Kintana upon request. The Kintana software and its documentation may not be sublicensed and may not be transferred without the prior written consent of Kintana.

Your right to copy Kintana software and this documentation is limited by copyright law. Making unauthorized copies, adaptations, or compilation works (except for archival purposes or as an essential step in the utilization of the program in conjunction with certain equipment) is prohibited and constitutes a punishable violation of the law.

THIS DOCUMENTATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. IN NO EVENT SHALL KINTANA BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OR DATA, INTERRUPTION OF BUSINESS, OR FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, ARISING FROM ANY ERROR IN THIS DOCUMENTATION.

Kintana may revise this documentation from time to time without notice.

Copyright © 1997, 1998, 1999, 2000, 2001, 2002, 2003 Kintana, Incorporated. All rights reserved.

Kintana, Kintana Deliver, Kintana Create, Kintana Drive, Kintana Dashboard, Kintana Accelerator, Kintana Demand Management (DM), Kintana Portfolio Management (PFM), Kintana Program Management Office (PMO), Kintana Enterprise Change Management (ECM), Object\*Migrator, GL\*Migrator and the Kintana logo are trademarks of Kintana, Incorporated. All other products or brand names mentioned in this document are the property of their respective owners.

Kintana Version 5.0.0

© Kintana, Incorporated 1997 - 2003

All rights reserved.

Printed in USA

**Kintana, Inc.**

1314 Chesapeake Terrace, Sunnyvale, California 94089

Telephone: (408) 543-4400

Fax: (408) 752-8460

<http://www.kintana.com>

# Contents

|   |           |
|---|-----------|
| <b>Chapter 1</b>                                    |           |
| <b>Introduction</b> .....                           | <b>5</b>  |
| <b>Who should read this guide</b> .....             | <b>5</b>  |
| <b>Additional Resources</b> .....                   | <b>6</b>  |
| Kintana Documentation .....                         | 6         |
| <i>Kintana Business Application Guides</i> .....    | 7         |
| <i>User Guides</i> .....                            | 7         |
| <i>Kintana Application Reference Guides</i> .....   | 7         |
| <i>Kintana Instance Administration Guides</i> ..... | 8         |
| <i>External System Integration Guides</i> .....     | 8         |
| <i>Kintana Solution Guides</i> .....                | 8         |
| <i>Kintana Accelerator Guides</i> .....             | 9         |
| Kintana Services .....                              | 9         |
| Kintana Education .....                             | 9         |
| Kintana Support .....                               | 10        |
| .....   | 10        |
| <b>Chapter 2</b>                                    |           |
| <b>Using Commands in Kintana</b> .....              | <b>11</b> |
| <b>Commands Overview</b> .....                      | <b>11</b> |
| Where Commands are Used .....                       | 11        |
| Commands Interface .....                            | 12        |
| Object Type Commands and Workflow .....             | 15        |
| Request Type Commands and Workflow .....            | 16        |
| Kintana Special Commands .....                      | 16        |
| <b>Commands Steps</b> .....                         | <b>17</b> |
| Kintana Command Language .....                      | 18        |
| <b>Command Conditions</b> .....                     | <b>18</b> |
| <b>Example Command Uses</b> .....                   | <b>19</b> |
| <b>Chapter 3</b>                                    |           |
| <b>Special Commands</b> .....                       | <b>21</b> |
| <b>Special Command Interface</b> .....              | <b>22</b> |
| Special Command Workbench .....                     | 22        |

|  |           |
|--|-----------|
| Special Command Window .....   | 23        |
| Special Command General Information Region .....                         | 23        |
| Parameters Tab .....   | 24        |
| Commands Tab .....   | 25        |
| <i>Command Conditions</i> .....  | 26        |
| <i>Parameters in Command Steps</i> .....                                 | 27        |
| <i>Special Command Builder</i> .....                                     | 29        |
| Ownership Tab .....  | 29        |
| Used By Tab .....  | 30        |
| <b>Creating and Editing Special Commands .....</b>                       | <b>31</b> |
| Creating a New Special Command .....                                     | 31        |
| Creating and Editing Special Command Parameters .....                    | 33        |
| Adding Parameters to Special Commands .....                              | 33        |
| Editing Special Command Parameters .....                                 | 35        |
| Deleting Parameters .....  | 35        |
| Setting Ownership for Special Commands .....                             | 36        |
| <b>Using Kintana System Special Commands .....</b>                       | <b>37</b> |
| Adding Special Commands to Command Steps Using the Command Builder ..... | 38        |
| Nesting Special Commands .....   | 39        |
| <br>   |           |
| <b>Chapter 4</b>   |           |
| <b>Using Tokens .....</b>  | <b>41</b> |
| <b>What are Tokens? .....</b>  | <b>41</b> |
| <b>Where Tokens Are Used .....</b>                                       | <b>42</b> |
| <b>Token Builder Window Overview .....</b>                               | <b>43</b> |
| <b>Token Formats .....</b>   | <b>45</b> |
| Default Format .....   | 47        |
| Explicit Entity Format .....   | 47        |
| User Data Format .....   | 50        |
| Parameter Format .....   | 51        |
| Request Field Tokens .....   | 52        |
| <i>Request Token Prefixes</i> .....                                      | 52        |
| <i>Tokens in Request Table Components</i> .....                          | 53        |
| Sub-Entity Format .....  | 55        |
| Environment and Environment Application Tokens .....                     | 56        |
| <b>Token Evaluation .....</b>  | <b>58</b> |
| <br>   |           |
| <b>Appendix A</b>  |           |
| <b>System Special Commands .....</b>                                     | <b>61</b> |
| <b>Special Commands in Kintana .....</b>                                 | <b>61</b> |
| ksc_connect Special Commands .....                                       | 62        |
| ksc_connect_dest_client .....  | 63        |
| <i>Example Using ksc_connect_dest_client</i> .....                       | 63        |

|  |    |
|--|----|
| ksc_connect_dest_server.....                                   | 63 |
| <i>Example using ksc_connect_dest_server</i> .....             | 64 |
| ksc_connect_source_client .....                                | 64 |
| <i>Example using ksc_connect_source_client</i> .....           | 65 |
| ksc_connect_source_server .....                                | 65 |
| <i>Examples using ksc_connect_source_server</i> .....          | 66 |
| ksc_exit .....   | 66 |
| ksc_copy Special Commands .....                                | 66 |
| ksc_copy_client_client .....                                   | 67 |
| <i>Example #1 using ksc_copy_client_client</i> .....           | 67 |
| <i>Example #2 using ksc_copy_client_client</i> .....           | 68 |
| ksc_copy_client_server .....                                   | 68 |
| <i>Example using ksc_copy_client_server</i> .....              | 68 |
| ksc_copy_server_client.....                                    | 69 |
| <i>Example using ksc_copy_server_client</i> .....              | 69 |
| ksc_copy_server_server.....                                    | 70 |
| <i>Example using ksc_copy_server_server</i> .....              | 70 |
| ksc_copy_client_tmp .....                                      | 71 |
| ksc_copy_server_tmp .....                                      | 71 |
| ksc_copy_tmp_client .....                                      | 72 |
| ksc_copy_tmp_server .....                                      | 72 |
| ksc_respond.....   | 73 |
| ksc_simple_respond.....  | 73 |
| <i>Examples using ksc_simple_respond</i> .....                 | 74 |
| ksc_local_exec .....   | 75 |
| <i>Example using ksc_local_exec</i> .....                      | 75 |
| ksc_replace.....   | 76 |
| <i>Example using ksc_replace</i> .....                         | 76 |
| ksc_set .....  | 76 |
| <i>Example using ksc_set</i> .....                             | 77 |
| ksc_set_env.....   | 77 |
| ksc_store .....  | 78 |
| <i>Example using ksc_store</i> .....                           | 78 |
| ksc_comment .....  | 79 |
| ksc_consub.....  | 79 |
| <i>Example using ksc_consub</i> .....                          | 79 |
| ksc_begin_script / ksc_end_script.....                         | 80 |
| <i>Example using ksc_begin_script and ksc_end_script</i> ..... | 81 |
| ksc_copy_script Special Commands .....                         | 82 |
| ksc_copy_script_dest_client .....                              | 82 |
| ksc_copy_script_dest_server .....                              | 82 |
| ksc_copy_script_source_client.....                             | 83 |
| ksc_copy_script_source_server.....                             | 83 |
| ksc_om_migrate.....  | 84 |
| <i>Example using ksc_om_migrate</i> .....                      | 85 |
| ksc_capture_output.....  | 85 |
| ksc_gl_migrate.....  | 86 |

---

|  |            |
|--|------------|
| Example ksc_gl_migrate.....                            | 86         |
| ksc_parse_jcl .....                                    | 87         |
| ksc_submit_job.....                                    | 87         |
| ksc_set_exit_value .....                               | 87         |
| ksc_clear_exit_value .....                             | 88         |
| ksc_run_sql.....                                       | 88         |
| Example ksc_run_sql.....                               | 89         |
| <b>Summary of All Special Command Parameters .....</b> | <b>89</b>  |
| <b>Appendix B</b>                                      |            |
| <b>Tokens .....</b>                                    | <b>95</b>  |
| <b>Kintana System Tokens.....</b>                      | <b>96</b>  |
| <b>Field Group Tokens .....</b>                        | <b>126</b> |

# Chapter 1 Introduction

Commands and Tokens are used throughout your Kintana implementation to enable advanced automation and defaulting.

Commands define the heart of the execution layer within your deployment system. Commands tell Kintana precisely which steps must be executed at a specific Workflow step. This can involve such activities as migrating a file, executing a script, performing some data analysis, or compiling code.

Tokens are variables that can be used to reference information that is undefined until the Kintana product is actually used a particular context. This includes such things as setting variables in Kintana commands or using tokens within Notifications to specify the recipients.

This document includes the following chapters:

- [\*Using Commands in Kintana\*](#)
- [\*Special Commands\*](#)
- [\*Using Tokens\*](#)
- [\*System Special Commands\*](#)
- [\*Tokens\*](#)

## Who should read this guide

This document provides information on using Kintana Commands and Tokens. This reference guide is used primarily by:

- Configuration experts configuring a deployment system.
- Configuration experts configuring a Request resolution system.

- Business modelers who need to modify the following Kintana entities: Workflows, Object Types, Request Types, Validations, and Report Types.



Users must have a Power license to access the screens and windows described in this document.

## Additional Resources

Kintana provides the following additional resources to help you successfully implement, configure, maintain and fully utilize your Kintana installation:

- [Kintana Documentation](#)
- [Kintana Services](#)
- [Kintana Education](#)
- [Kintana Support](#)

### *Kintana Documentation*

Kintana product documentation is linked from the Kintana Library page. This page is accessed by:

- Selecting **HELP > KINTANA LIBRARY** from the Kintana Workbench menu.
- Selecting **HELP > CONTENTS AND INDEX** from the menu bar on the HTML interface. You can then click the **KINTANA LIBRARY** link to load the full list of product documents.

Kintana organizes their documents into a number of user-based categories. The following section defines the document categories and lists the documents currently available in each category.

- [Kintana Business Application Guides](#)
- [User Guides](#)
- [Kintana Application Reference Guides](#)
- [Kintana Instance Administration Guides](#)
- [External System Integration Guides:](#)



- [Kintana Solution Guides](#)
- [Kintana Accelerator Guides](#)

## **Kintana Business Application Guides**

Provides instructions for modeling your business processes in Kintana. These documents contain process overviews, implementation instructions, and detailed examples.

- Configuring a Request Resolution System (Create)
- Configuring a Deployment and Distribution System (Deliver)
- Configuring a Release Management System
- Configuring the Kintana Dashboard
- Managing Your Resources with Kintana
- Kintana Reports

## **User Guides**

Provides end-user instructions for using the Kintana products. These documents contain comprehensive processing instructions.

- Processing Packages (Deliver) User Guide
- Processing Requests (Create) User Guide
- Processing Projects (Drive) User Guide
- Navigating the Kintana Workbench:  
Provides an overview of using the Kintana Workbench
- Navigating Kintana:  
Provides an overview of using the Kintana (HTML) interface

## **Kintana Application Reference Guides**

Provides detailed reference information on other screen groups in the Kintana Workbench. Also provides overviews of Kintana's command usage and security model.

- Reference: Using Commands in Kintana
- Reference: Kintana Security Model
  
- Workbench Reference: Deliver
- Workbench Reference: Configuration
- Workbench Reference: Create
- Workbench Reference: Dashboard
- Workbench Reference: Sys Admin
- Workbench Reference: Drive
- Workbench Reference: Environments

### **Kintana Instance Administration Guides**

Provides instructions for administrating the Kintana instances at your site. These documents include information on user licensing and archiving your Kintana configuration data.

- Kintana Migration
- Kintana Licensing and Security Model

### **External System Integration Guides:**

Provides information on how to use Kintana's open interface (API) to access data in other systems. Also discusses Kintana's Reporting meta-layer which can be used by third party reporting tools to access and report on Kintana data.

- Kintana Open Interface

### **Kintana Solution Guides**

Provides information on how to configure and use functionality associated with the Kintana Solutions. Each Kintana Solution provides a User Guide for instructions on end-use and a Configuration Guide for instructions on installing and configuring the Solution.

## Kintana Accelerator Guides

Provides information on how to configure and use the functionality associated with each Kintana Accelerator. Kintana Accelerator documents are only provided to customers who have purchased a site-license for that Accelerator.



Note

Kintana provides documentation updates in the Download Center section of the Kintana Web site ([http://www.kintana.com/support/download/download\\_center.htm](http://www.kintana.com/support/download/download_center.htm)).

A username and password is required to access the Download Center. These were given to your Kintana administrator at the time of product purchase. Contact your administrator for information on Kintana documentation or software updates.

## Kintana Services

Kintana is a strategic partner to its clients, assisting them in all aspects of implementing a Kintana technology chain - from pilot project to full implementation, education, project turnover, and ongoing support. Our Total Services Model tailors solution and service delivery to specific customer needs, while drawing on our own knowledgebank and best practices repository. Learn more about Kintana Services from our Web site:

<http://www.kintana.com/services/services.shtml>

## Kintana Education

Kintana has created a complete product training curriculum to help you achieve optimal results from your Kintana applications. Learn more about our Education offering from our Web site:

<http://www.kintana.com/services/education/index.shtml>

## *Kintana Support*

Kintana provides web-based interactive support for all products in the Kintana product suite via Contori.

<http://www.contori.com>

Login to Contori to enter and track your support issue through our quick and easy resolution system. To log in to Contori you will need a valid email address at your company and a password that will be set by you when you register at Contori.

**Chapter**  
**2****Using Commands in Kintana**

The following sections provide an overview and examples for using commands in Kintana.

- *Commands Overview*
- *Kintana Command Language*
- *Kintana Special Commands*
- *Commands Steps*
- *Command Conditions*
- *Example Command Uses*

## Commands Overview

Commands define the heart of the execution layer within your deployment or Request resolution system. Commands tell Kintana precisely which steps must be executed at a specific Workflow step. This can involve such activities as migrating a file, executing a script, performing some data analysis, updating field information, or compiling code.

## Where Commands are Used

Commands are used in the following Kintana entities to enhance your implementation and enable sophisticated command-line automation:

- Object Types
- Request Types

- Report Types
- Workflows
- Validations

## Commands Interface

Commands are accessible through the **COMMANDS** tab of the Object Type, Request Type, Report Type, Validation, Workflow Step Source, or Special Command screens and consist of command information and command steps. In this chapter, the examples are accessed through the Deliver: Object Types screen, but the interface is the same in other screens where commands are configured.

Double-click the Command Step to open the EDIT COMMAND window. The EDIT COMMAND window displays the shell script code in the STEPS window, as shown in *Figure 2-1*.

Double click the Command to open the Edit Command window.

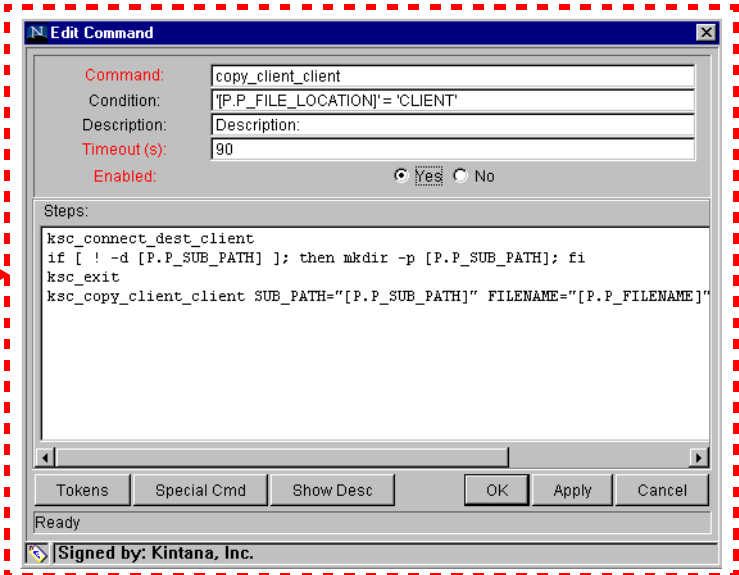
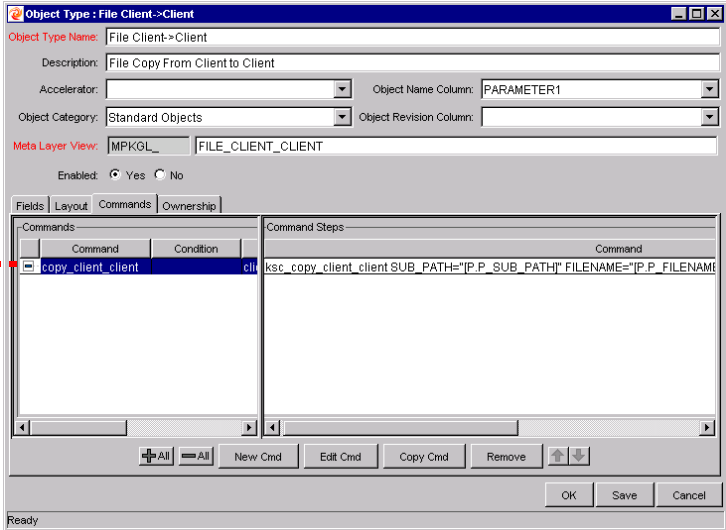


Figure 2-1 Commands Tab and Edit Command Window

To generate a new command, click **NEW CMD** in the **COMMANDS** tab. This opens the **NEW COMMAND** window shown in *Figure 2-2*. *Table 2-1* shows the fields included in this window.

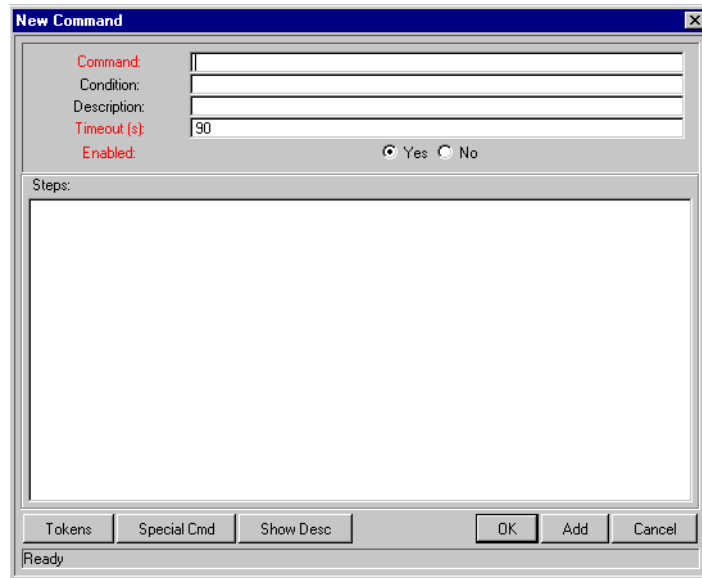


Figure 2-2 New Command Window

Table 2-1. New Command Window Fields

| Field       | Description   |
|-------------|---|
| Command     | A simple name for the command.  |
| Condition   | A condition that determines whether the steps for the command are executed or not. (See <i>“Command Conditions”</i> on page 18 below for more information).                             |
| Description | A description of the command.   |
| Timeout     | The amount of time the command will be allowed to run before its process is terminated. This mechanism is used to abort commands that are hanging or taking an abnormal amount of time. |
| Enabled?    | Determines whether the command is enabled for execution.  |

Each Object Type, Request Type, Validation, Workflow step source, or Report Type may have many commands, and each command may have many command steps. A command may be viewed as a particular function for an object. Copying a file may be one command, and checking that file into version control may be another. To perform these functions, a series of events needs to take place, and these events are defined in the command steps.



---

An additional level of flexibility is introduced when some commands must only be executed in certain cases. This is powered by the condition field of the commands and is discussed in “[Command Conditions](#)” on page 18.

Note

The Timeout value is divided evenly between each command step included in the command. Therefore, you should configure the timeout value so that it allows enough time for each command step to execute. For example, Command ABC is comprised of 10 command steps, and has a timeout of 1,000 seconds. Each command step is therefore allotted 100 seconds to execute. If any command step takes longer than 100 seconds, the command times-out.

## Object Type Commands and Workflow

Object Type Commands are tightly integrated with the Kintana Workflow engine. The commands contained in an Object Type are executed at EXECUTION Workflow steps in Deliver Package Lines.

It is important to note the following concepts regarding Command/Workflow interaction:

- To execute Object Type commands at a particular Workflow step, the Workflow step must be configured with the following parameters:
  - Workflow step must be an Execution type step.
  - WORKFLOW SCOPE = **PACKAGES**.
  - EXECUTION TYPE = **BUILT-IN WORKFLOW EVENT**.
  - WORKFLOW COMMAND = **EXECUTE\_OBJECT\_COMMANDS**.
- When the object reaches the Workflow step (with WORKFLOW COMMAND = **EXECUTE\_OBJECT\_COMMANDS**), all of the Object Type commands whose conditions are satisfied will be run in the order they are entered in the Object Type’s command panel.
- The Object Type can be configured to run only certain commands at a particular step. To do this, specify command conditions. See “[Command Conditions](#)” on page 18 for details.

## Request Type Commands and Workflow

Similar to Object Type commands, Request Type commands define the execution layer within Kintana Create. While most of the resolution process for a Request is analytically based, cases may arise for specific Request Types where system changes are required. In these cases, Request Type commands can be used to automatically perform these changes.

Request Type Commands are tightly integrated with the Kintana Workflow engine. The commands contained in a Request Type are executed at EXECUTION Workflow steps.

It is important to note the following concepts regarding Command/Workflow interaction:

- To execute Request Type commands at a particular Workflow step, the Workflow step must be configured with the following parameters:
  - Workflow step must be an Execution type step.
  - WORKFLOW SCOPE = **REQUESTS**
  - EXECUTION TYPE = **BUILT-IN WORKFLOW EVENT**.
  - WORKFLOW COMMAND = **EXECUTE\_REQUEST\_COMMANDS**.
- When the Request reaches the Workflow step (with WORKFLOW COMMAND = **EXECUTE\_REQUEST\_COMMANDS**), all of the commands whose conditions are satisfied will be run in the order they are entered in the Request Type's command panel.
- The Request Type can be configured to run only certain commands at a particular step. To do this, specify command conditions. See "[Command Conditions](#)" on page 18 for details.

## Kintana Special Commands

Kintana Object Types, Request Types, Report Types, Workflows and Validations all use commands to access the Kintana execution layer. In order to simplify the use of command executions, Kintana contains a predefined set of Special Commands. Users can also create their own Special Commands.

Special Commands are commands with variable parameters and are used in Object Type, Request Type, Report Type, Workflow, and Validation command steps. These command steps perform a variety of functions, such as

copying files between environments and establishing connections to environments for remote command execution. Kintana features two types of Special Commands:

- System Special Commands - These commands are shipped with Kintana. System Special Commands are read-only and have the naming convention “ksc\_command\_name.” System Special Commands always begin with “ksc\_.”
- User Defined Special Commands - These commands are user-defined and have the naming convention “sc\_command\_name.” User-defined Special Commands must begin with “sc\_.”

Kintana Special Commands act as sub-programs that can be reused where ever needed. It is often more convenient to create a Special Command for a program that will be used in multiple places rather than placing the individual commands into every Object Type, Request Type, etc. that need them.

## Commands Steps

Command steps represent the actual directives that Kintana specifies to execute the commands. A command step can be an actual command-line directive that is sent to the Kintana server or target machine or can be one of Kintana’s many “special commands.” [Table 2-2](#) describes the fields in the COMMAND STEPS region of the NEW/EDIT COMMANDS dialog.

Table 2-2. Command Steps

| Field       | Description   |
|-------------|---|
| Steps       | Defines the command-line directive or special command to be issued. |
| Description | Describes each of the command steps.                                |



Note

The Kintana Execution Engine will execute the commands and command steps in the order they are displayed in the **COMMANDS** tab. To change the order of the commands or the command steps, in the **COMMANDS** tab, select the given command or command step and use the arrow buttons to move the selected item.

## Kintana Command Language

The command steps in a command define the actual system-level executions that need to be performed to achieve the desired function of the command. Command steps can be UNIX commands, third party application commands, or Kintana Special Commands. Special commands are reusable routines that are defined in Kintana. Kintana also supplies a number of system special commands that are used to perform common execution events (connecting to environments, copying files, etc.). Kintana tokens can be used within command steps.

## Command Conditions

In many situations, it may be necessary to run a different set of commands depending on the context of execution. This flexibility is achieved through the use of conditional commands. The `CONDITION` field for a command is used to define the situation under which the associated command steps execute.

Conditions are evaluated as boolean expressions. If the expression evaluates to true, the command is executed. If false, the command is skipped and the next command is evaluated. If no condition is specified, the command is always executed. The syntax of a condition is identical to the “where” clause of a SQL statement, which allows enormous flexibility when evaluating scenarios. Some example conditions are detailed in the following table:

*Table 2-3. Example Conditions*

| <b>Condition</b>                          | <b>Evaluates to</b>   |
|---|---|
| BLANK                                     | Command will be executed in all situations.   |
| '[P.P_VERSION_LABEL]' IS NOT NULL         | Command will be executed if the parameter with the token P_VERSION_LABEL in the Package line is not null. |
| '[DEST_ENV.ENVIRONMENT_NAME]' = 'Archive' | Command will be executed when the destination environment is named “Archive”.                             |
| '[AS.SERVER_TYPE_CODE]' = 'UNIX'          | Command will be executed if the application server is installed on a UNIX machine.                        |



Don't forget to place single quotes around string literals or tokens that will evaluate strings.

The condition can include Tokens. See [“Using Tokens”](#) on page 41 for more information.

## Example Command Uses

This section provides a number of operations that you can execute using commands. Sample code for configuring many of these cases is included in the [“System Special Commands”](#) on page 61.

- Commands for connecting to machines.
  - o Connect to the destination environment and run system commands
  - o Connect to an alternate environment and run command (Environment override)
- Commands for manipulating data. (Kintana fields and other info stored in files or database)
  - o Set a value in a Package Line
  - o Create, run and delete a script
  - o Extract information from a file (version number)
- Commands for running operating system-specific commands. (NT and Unix)
  - o Starting a server
  - o Stopping a server
- Commands for running program-specific commands
  - o Checking files in and out of a Version control system
- Commands for copying files



# Chapter 3 Special Commands

Kintana Object Types, Request Types, Report Types, Workflows and Validations all use commands to access the Kintana execution layer. In order to simplify the use of command executions, Kintana contains a predefined set of Special Commands. Users can also create their own Special Commands.

Special Commands are commands with variable parameters and are used in Object Types, Request Types, Report Types, Workflows, and Validation command steps. (Workflows use Special Commands in their Workflow Step Sources.) These command steps perform a variety of functions, such as copying files between environments and establishing connections to environments for remote command execution. Kintana features two types of Special Commands:

- System Special Commands - These commands are shipped with the Kintana Product Suite. System Special Commands are read-only and have the naming convention “ksc\_command\_name.” System Special Commands always begin with “ksc\_.”
- User Defined Special Commands - These commands are user-defined and have the naming convention “sc\_command\_name.” User-defined Special Commands must begin with “sc\_.”

This chapter discusses the interface for creating, editing and using Special Commands in the Kintana Product Suite. The following topics are discussed:

- [“Special Command Interface”](#) on page 22
- [“Creating and Editing Special Commands”](#) on page 31
- [“Using Kintana System Special Commands”](#) on page 37

See the [“System Special Commands”](#) chapter on page 61 for a detailed description of System Special Commands and their parameters.

## Special Command Interface

The Special Command interface, shown in [Figure 3-1](#), is used to create, view and edit Special Commands. The Special Command interface consists of the SPECIAL COMMAND WORKBENCH and SPECIAL COMMAND window. To access the Special Command interface, click **CONFIGURATION** in the shortcut bar and click the **SPECIAL COMMANDS** icon.

## Special Command Workbench

The SPECIAL COMMAND WORKBENCH lets you search for a particular Special Command in the **QUERY** tab using the following criteria:

- **SPECIAL COMMAND NAME** - Filter for Special Commands where the name matches a given string.
- **DESCRIPTION** - Filter for Special Commands where the description matches a given string.
- **ENABLED** - Filter for Special Commands that are enabled or disabled.

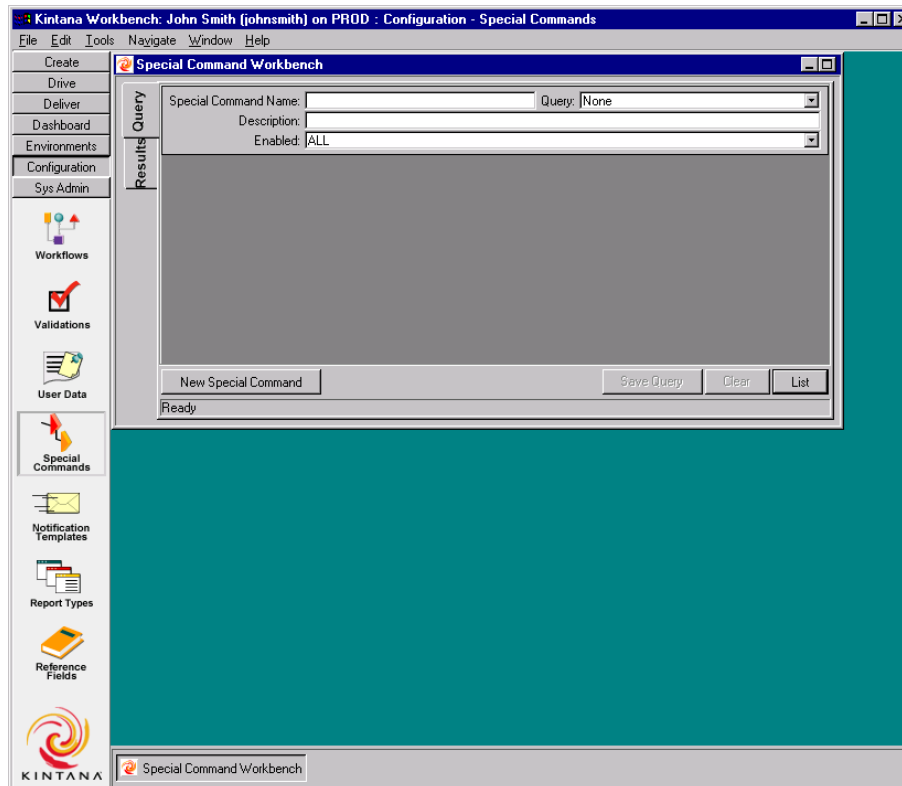


Figure 3-1 Special Command Workbench



# Special Command Window

The SPECIAL COMMAND window, shown in *Figure 3-2*, is used to define and configure Kintana Special Commands. It consists of the following regions: the Special Command general information region, the **PARAMETERS** tab, the **COMMANDS** tab, the **OWNERSHIP** tab, and the **USED BY** tab.

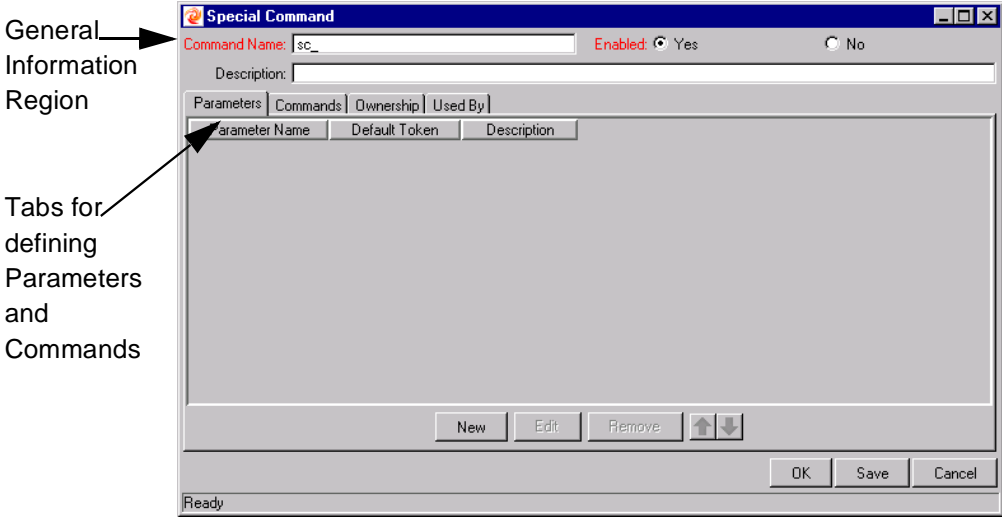


Figure 3-2 Special Command Window

## Special Command General Information Region

The Special Command general information region displays the basic header information for the Special Commands. It consists of the fields described in *Table 3-1*.

*Table 3-1. Special Commands Information Fields*

| Name         | Field    |                     | Description   |
|--------------|----------|---------------------|---|
|              | Required | Type                |   |
| COMMAND NAME | Y        | Text Field          | The name of the Special Command. This can only be updated when generating or editing a user-defined Special Command.                      |
| ENABLED?     | Y        | Yes/No Radio Button | Determines whether or not the Special Command is enabled for use in Workflows, Object Types, Report Types, Request Types and Validations. |
| DESCRIPTION  | N        | Text Field          | A description of the Special Command. This can only be updated when generating or editing a user-defined Special Command.                 |

### *Parameters Tab*

The **PARAMETERS** tab displays the current parameters for the Special Command. Most Special Commands have parameters to override standard behavior. Nearly all parameters are optional. When a parameter is not passed to a Special Command and the default value for the parameter is a custom Token, the entity using the command must contain a field with that Token.

## Example

The 'KSC\_COPY\_SERVER\_SERVER' Special Command shown in this example is used in an Object Type. The parameter FILENAME is not specified and defaults to [P.P\_FILENAME] because it is not explicitly passed.

```
ksc_copy_server_server
```

This makes 'KSC\_COPY\_SERVER\_SERVER' equivalent to:

```
ksc_copy_server_server FILENAME=" [P.P_FILENAME] "
```

because "[P.P\_FILENAME]" is the default token for the parameter FILENAME. The command execution engine evaluates the token [P.P\_FILENAME] so it must be defined for the entity (the specific Object Type, Report Type or Request Type).

To override the default token, pass in another value for the parameter. A few examples are:

```
ksc_copy_server_server FILENAME="document.txt"
ksc_copy_server_server FILENAME=" [P.DOCUMENT_NAME] "
```

This method of passing parameters is explained in more detail in the section entitled "*Special Command Builder*" on page 29.

## Note

Custom Tokens are defined for specific Object Types, Request Types, and Report Types, and are referenced using the '[P.TOKEN\_NAME]' syntax. See "*System Special Commands*" on page 61 for a list of all predefined Special Command parameters and their default Tokens.

## Commands Tab

The **COMMANDS** tab lets you define and configure the commands and command steps used by each user-defined Special Command. It is also possible to view the command information for the predefined Kintana system Special Commands.

Kintana commands are designed to have a similar look-and-feel to the UNIX and DOS operating system command structure. The specific parts of a command, the command steps, are often just command-prompt directives.

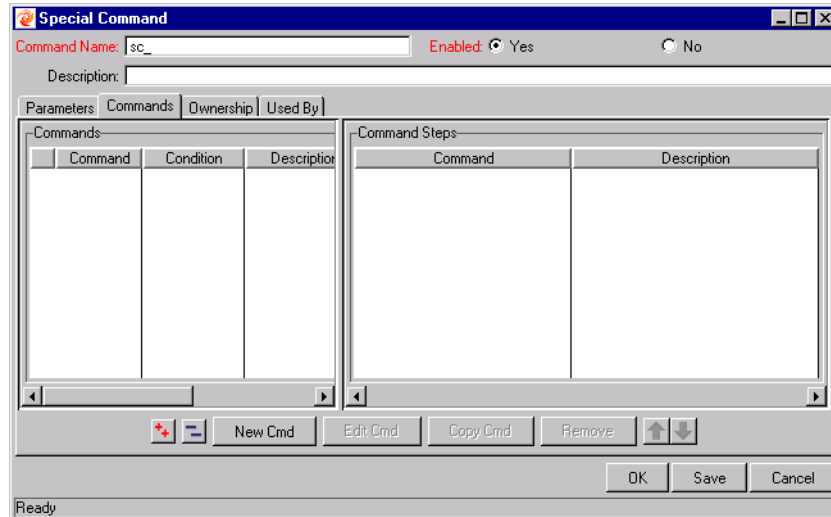


Figure 3-3 Special Commands - Commands Tab

Commands are accessible through the **COMMANDS** tab of the SPECIAL COMMANDS window and consist of command information and command steps.

### Command Conditions

In many situations, it may be necessary to run a different set of commands depending on the context of execution. For example, one command may be needed to update a Web page, while another command may be required to set-up an account on the Sales Automation application.

This flexibility is achieved through the use of conditional commands. The Condition field for an object command provides the ability to define the situation under which the associated command steps will execute.

Conditions are evaluated as Boolean expressions. If the expression evaluates to TRUE, the command is executed. If FALSE, the command is skipped and the next command is evaluated to see if it should run. If no condition is specified the command is always executed. The syntax of a condition is identical to the WHERE clause of a SQL statement, which allows flexibility when evaluating scenarios. Some example conditions are given in [Table 3-2](#).

Table 3-2. Example Conditions

| Condition | Evaluates to                        |
|-----------|-------------------------------------|
| BLANK     | Command executes in all situations. |

Table 3-2. Example Conditions

| Condition                    | Evaluates to   |
|------------------------------|--|
| '[REQ.DEPARTMENT]' = 'SALES' | Command executes when the department for the Request is named SALES. |
| '[REQ.PRIORITY]' = 'HIGH'    | Command executes if the priority assigned to the Request is HIGH.    |



Note

When using conditional commands, strings must be enclosed by single quotes.

The condition can include a Token. See [“Using Tokens”](#) on page 41 for more information.

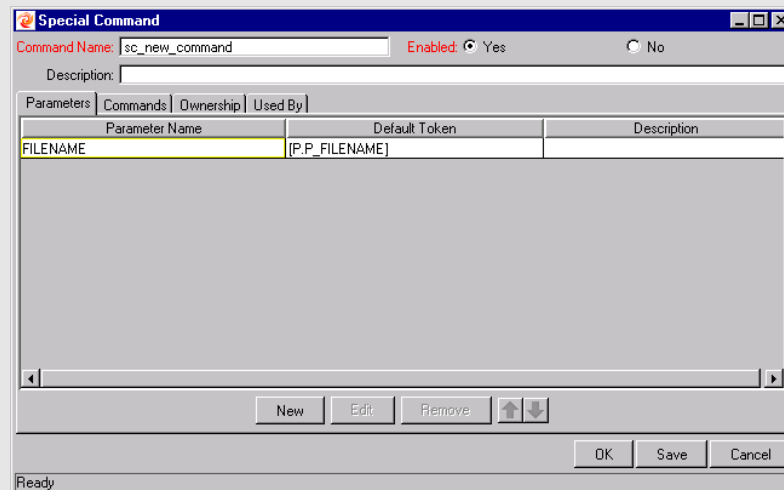
### Parameters in Command Steps

In the command steps within a Special Command, parameters are referred to as their default Tokens. When the Special Command is executed with a value specified for a parameter, this value will replace the default token throughout the Special Command steps.



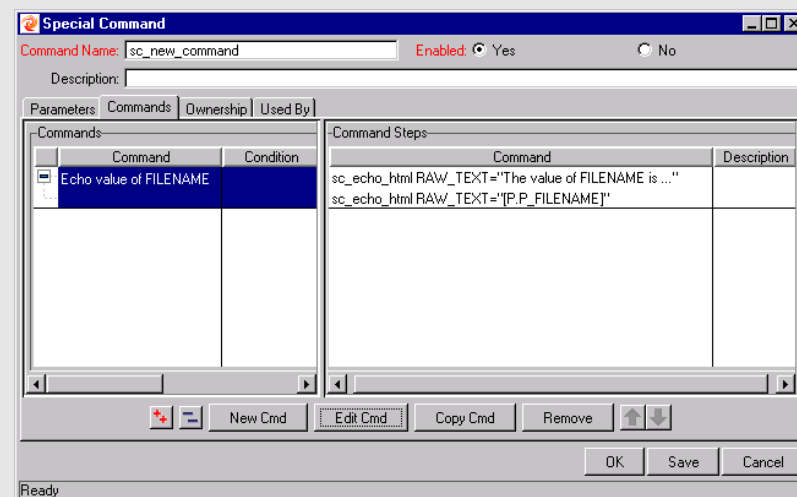
Example

We have created a Special Command to echo a string as an HTML tag, called `sc_echo_html`. This Special Command takes the parameter `RAW_TEXT`. We would like to use `sc_echo_html` in another Special Command, `sc_new_command`, to echo the parameter value `FILENAME`. `FILENAME` has a default token of `[P.P_FILENAME]`.



To accomplish this, the following command steps are entered in a command for `sc_new_command`:

```
sc_echo_html RAW_TEXT="The value of FILENAME is..."
sc_echo_html RAW_TEXT="[P.P_FILENAME]"
```



Note that the command step uses the default token to refer to the value of the Special Command parameter. The parameter name is only used when invoking a Special Command. From an Object Type, Request Type, etc.



Parameters cannot be used in command conditionals.

Continuing from the previous example, suppose that we have a Special Command with the parameter FILENAME, whose default token is [P.P\_FILENAME]. In command conditionals, the token [P.P\_FILENAME] will always be evaluated normally, regardless of whether our Special Command was called with a value for the parameter FILENAME.

### Special Command Builder

The Special Command Builder is a tool designed to simplify the use of Special Commands by ensuring proper formatting of the Command Step. The Special Command Builder, shown in *Figure 3-4*, is an interface where a Special Command can be selected and appropriate parameters can be entered. The Special Command builder outputs a line of text to the COMMAND field which can be used as a command step.

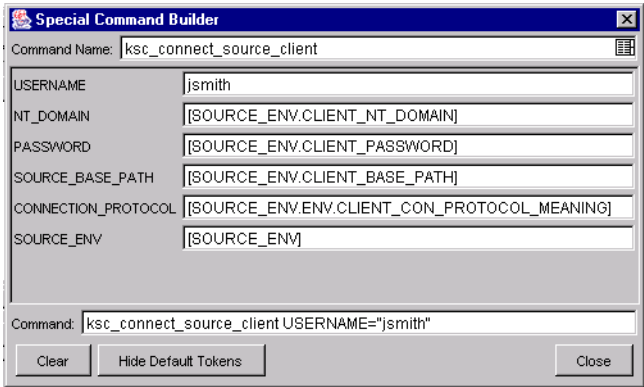


Figure 3-4 Special Command Builder

### Ownership Tab

The **OWNERSHIP** tab is used to select Ownership Groups for a specific Special Command. Members of Ownership Groups are the only users who have the right to edit, copy or delete this Special Command. This tab also displays Ownership Groups that have been linked to this entity. Ownership Groups can be deleted from this tab by selecting them and clicking **REMOVE**.

See *“Setting Ownership for Special Commands”* on page 36 for more information about setting Ownership for a new or existing Special Command.

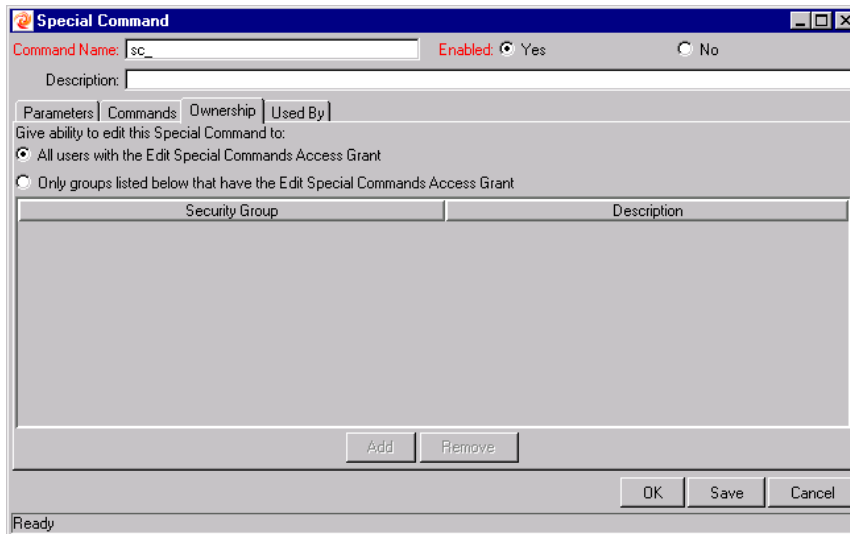


Figure 3-5 Ownership Tab

Table 3-3. Ownership tab fields

| Field  |              | Description   |
|--|--------------|---|
| Name   | Type         |   |
| All users with the Edit Special Command Access Grant                     | Radio Button | Enables all users with the EDIT SPECIAL COMMAND Access Grant to copy, edit and delete the Special Command.            |
| Only Groups listed below that have the Edit Special Command Access Grant | Radio Button | Limits the users who can copy, edit and delete the Special Command to members of the group listed in the below panel. |
| Add  | Button       | Click to add a new Security Group to the <b>OWNERSHIP</b> tab.  |
| Remove   | Button       | Click to remove selected Security Groups from the <b>OWNERSHIP</b> tab.   |

### Used By Tab

Click the **Used By** tab to view a list of entities that currently refer to the selected Special Command.



## Creating and Editing Special Commands

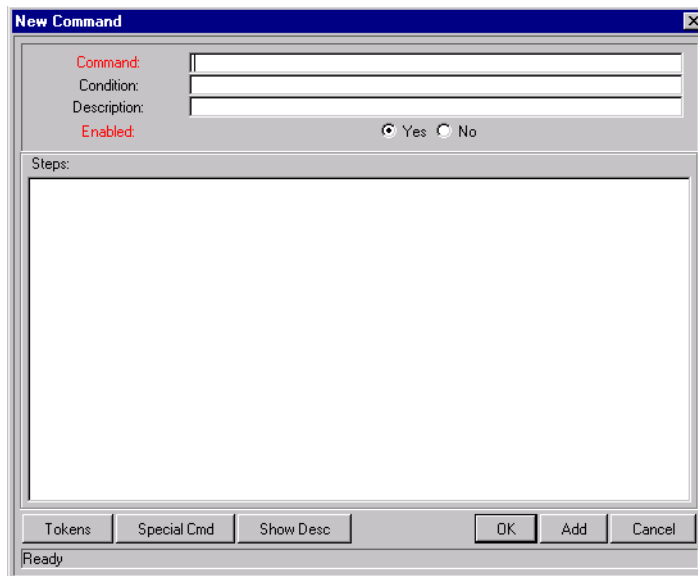
Kintana users can create their own Special Commands. This section describes the following procedures for working with Special Commands:

- [Creating a New Special Command](#)
- [Creating and Editing Special Command Parameters](#)
- [Adding Special Commands to Command Steps Using the Command Builder](#)

### Creating a New Special Command

To create a new Special Command:

1. From the SPECIAL COMMAND WORKBENCH, click **NEW SPECIAL COMMAND**. The SPECIAL COMMAND window opens.
2. Click the **COMMANDS** tab.
3. Click **NEW CMD**. The NEW COMMAND window opens. This window's fields are defined in [Table 3-4](#).



4. Enter information in the COMMAND, CONDITION and DESCRIPTION fields. See [“Command Conditions”](#) on page 26 for more details about defining Conditions.

5. The **ENABLED** radio button should be set to **YES**.
6. To add Tokens to the new Special Command, click **TOKENS**. The **TOKEN BUILDER** window opens.
  - a. Copy a Token from the **TOKEN BUILDER** window.
  - b. Paste it into the **NEW COMMAND** window's **STEPS** text area.
7. To use another Special Command in the Special Command you are defining, click **SPECIAL CMD**. The **SPECIAL COMMAND BUILDER** window opens. Select a Special Command from the **Command Name** field and enter any required parameters.
  - a. Copy the Special Command from the **SPECIAL COMMAND BUILDER** window.
  - b. Paste it into the **NEW COMMAND** window's **STEPS** text area.
8. Click **ADD** to add the new command to the **COMMAND** tab of the **SPECIAL COMMAND** window without closing the **NEW COMMAND** window.
9. Click **OK** to add the new command to the **COMMAND** tab of the **SPECIAL COMMAND** window and close the **NEW COMMAND** window.

The new Special Command has been created.

10. Click **SAVE** to save the new Special Command.

*Table 3-4. New Command Window Fields*

| Name        | Field    |            | Description  |
|-------------|----------|------------|--|
|             | Required | Type       |  |
| COMMAND     | Y        | Text Field | The name of the command.   |
| CONDITION   | N        | Text Field | A condition that determines whether the Command steps for the command are executed or not. (See " <a href="#">Command Conditions</a> " on page 26 for more information). |
| DESCRIPTION | N        | Text Field | A description of the command.  |

Table 3-4. New Command Window Fields

| Field    |          |                           | Description  |
|----------|----------|---------------------------|--|
| Name     | Required | Type                      |  |
| ENABLED? | Y        | Yes/No<br>Radio<br>Button | Determines whether the command is enabled for execution. |

## Creating and Editing Special Command Parameters

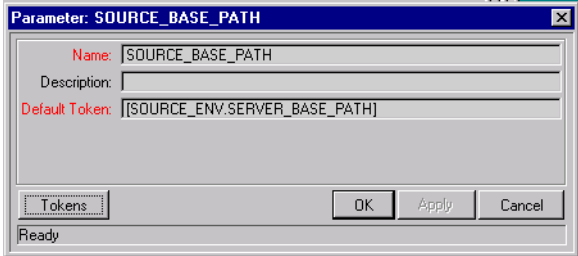
The following sections provide instructions for:

- [Adding Parameters to Special Commands](#)
- [Editing Special Command Parameters](#)
- [Deleting Parameters](#)

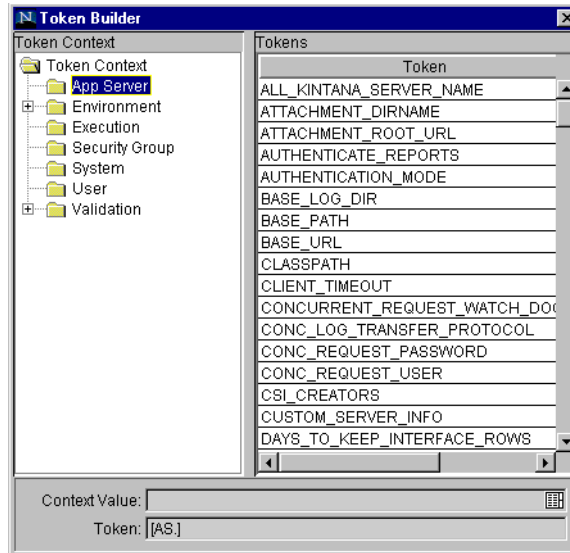
### Adding Parameters to Special Commands

To add a new parameter to a user-defined Special Command:

1. In the **PARAMETERS** tab of the SPECIAL COMMAND window, click **NEW**. The PARAMETER window opens. This window's fields are defined in [Table 3-5](#).



2. Fill in the NAME, DESCRIPTION and DEFAULT TOKEN fields. If you want to select an existing global token, follow Steps 4 through 10. If you manually entered a Token name in the Default Token field, go to Step 8.
3. To select an existing global token, click **TOKENS**. The TOKEN BUILDER window opens.



4. Open one of the folders in the TOKEN CONTEXT pane of the window. The available tokens for each folder display in the TOKENS pane of the window.
5. Select a token under the TOKEN column. When a token is selected, it enables the TOKEN field and displays the name of the selected token (including its prefix).
6. Copy the name of the token by first selecting it in the TOKEN field and then pressing Ctrl+C on your keyboard.
7. In the PARAMETER window, paste the token name into the DEFAULT TOKEN field by pressing Ctrl+V on your keyboard.
8. Click **OK** to add the field to the **PARAMETERS** tab and close the PARAMETER window.
9. Click **ADD** to add the field to the **PARAMETERS** tab without closing the PARAMETER window.

Table 3-5. Parameters Tab Fields

| Name             | Field    |            | Description   |
|------------------|----------|------------|---|
|                  | Required | Type       |   |
| NAME             | Y        | Text Field | The name of the parameter.  |
| DESCRIPTION      | N        | Text Field | A brief description of the parameter.   |
| DEFAULT<br>TOKEN | Y        | Text Field | This is the default token that used in the command unless otherwise specified. It must be unique for a Special Command. |

### Editing Special Command Parameters

To edit an existing parameter:

1. Open the Special Command.
2. In the **PARAMETERS** tab, double-click the Parameter. The **PARAMETER** window opens.
3. Make the desired changes in the **PARAMETER** window.
4. Click **APPLY** to apply the changes without closing the **PARAMETER** window.
5. Click **OK** to apply the changes and close the **PARAMETER** window.



Note

The parameter order can be altered by selecting a parameter in the **PARAMETERS** tab and clicking either the **UP** or **DOWN** arrow.

Changes to parameters already used by existing Request Types, Object Types, or Report Types can affect the way these entities function.

### Deleting Parameters

To delete a parameter:

1. Open the Special Command.
2. Select the parameter in the **PARAMETERS** tab.
3. Click **REMOVE**.

4. Click **OK** to save the information and close the SPECIAL COMMAND window.
5. Click **SAVE** to save the information without closing the SPECIAL COMMAND window.

The parameter is deleted from the Special Command.

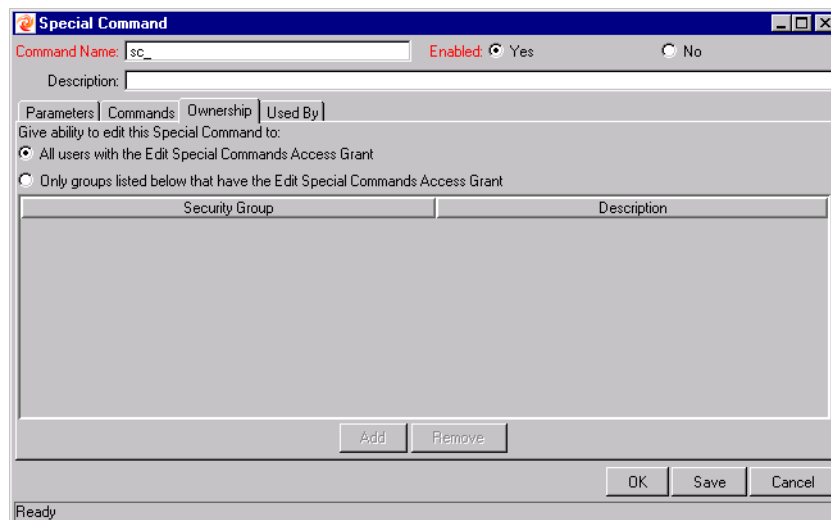
## Setting Ownership for Special Commands

Different groups of Kintana users can have exclusive control over the Special Commands used by their group. These groups are referred to as Ownership Groups. Members of the ownership group are the only users who can edit, delete or copy the Special Commands. Each Special Command can be assigned multiple ownership groups.

Ownership groups are defined in the SECURITY GROUP window in the Kintana Workbench. Refer to the "[Kintana Security Model](#)" for instructions on setting up Security Groups.

To set the Ownership for a Special Command:

1. Open the Special Commands window.
2. Click the **OWNERSHIP** tab.



3. Select the **ONLY GROUPS LISTED BELOW THAT HAVE THE EDIT SPECIAL COMMANDS ACCESS GRANT** option.

4. Click **ADD**. The ADD SECURITY GROUPS window opens.
5. Select a Security Group from the SECURITY GROUP auto-complete list.
6. Click **OK** to close the ADD SECURITY GROUP window. The Security Groups you selected display in the **OWNERSHIP** tab under the Security Group column.
7. Click **OK** in the SPECIAL COMMAND window to save the changes and close the window. Click **SAVE** to save the selection and leave the SPECIAL COMMAND window open.

Now only members of the Security Group(s) specified in the **OWNERSHIP** tab can edit, delete or copy this Special Command.

Note

If no Ownership groups are associated with the entity, the entity is considered global and any user with the Edit Access Grant for the entity can edit, copy or delete it. Refer to the "*Kintana Security Model*" for more information on Kintana Access Grants.

By default, Kintana administrators have the 'Ownership Override' access grant and can access configuration entities even if the administrator is not a member of one of the Ownership Groups and does not have the Edit Access Grant.

If a Security Group is disabled or loses the Edit Access Grant, that group will no longer have edit access for the entity.

## Using Kintana System Special Commands

Special Commands are added to Command Steps directly in the Kintana entity windows (Object Types, Request Types, Report Types, Validations and Workflows). For example, *Figure 3-6* shows an Object Type that has been generated using a combination of Special Commands.

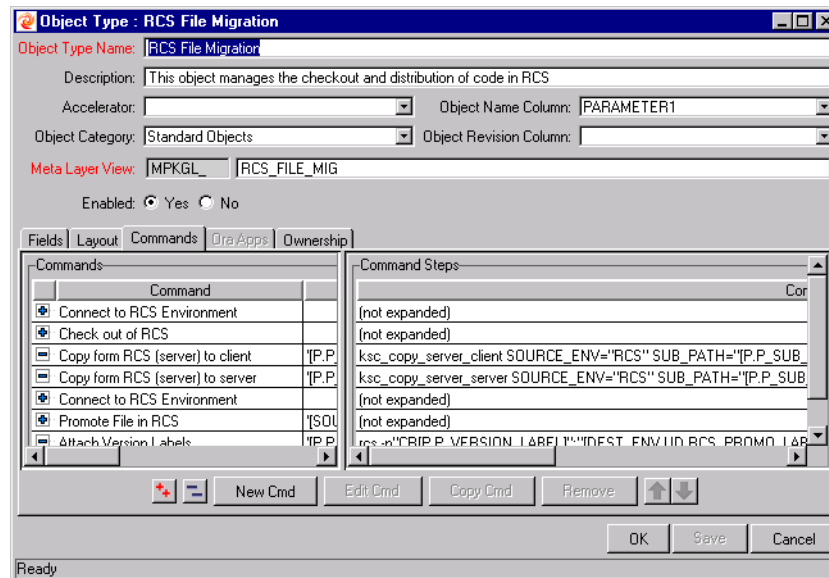


Figure 3-6 RCS File Migration Object Type

## Adding Special Commands to Command Steps Using the Command Builder

Special Commands can be added to any set of command steps in Object Types, Request Types, Report Types, Validations, Workflow Step sources, and other Special Commands. The Special Command Builder is available in the **COMMANDS** tab for each of these entities.

To build a command step using the Special Command Builder:

1. Go to the **COMMANDS** tab for the entity to which you are adding commands.
2. Click **NEW CMD** or edit an existing command. The **COMMAND** window opens.
3. Click **SPECIAL CMD**. The **SPECIAL COMMAND BUILDER** window opens.
4. Enter the a command name in the **COMMAND NAME** field or select it from the auto-complete list.

When you select a command name from the auto-complete list, its parameters appear in the **SPECIAL COMMAND BUILDER**.



Note

Both predefined (ksc\_command) and user defined (sc\_command) Special Commands can be used to build the command steps line. For more information on generating special commands, see “[Special Command Interface](#)” on page 22.

5. Replace the associated default token value with any desired parameter information. View the default tokens by clicking **SHOW DEFAULT TOKENS**. Hide the default tokens by clicking **HIDE DEFAULT TOKENS**.
6. When the parameters have been modified, select the text in the **COMMAND** field. Press Ctrl+C on your keyboard to copy the formatted Special Command.
7. Click **CLOSE** to close the **SPECIAL COMMAND BUILDER** window.
8. Click in the Steps text area of the **NEW COMMAND** window and press Ctrl+V on your keyboard to paste the Special Command step.
9. Fill in the remaining fields in the **NEW COMMAND** window.
10. The **ENABLED** radio button should be set to **YES**.
11. Click **OK** to add the command step to the **COMMAND** tab.

The new Special Command is now ready to be used in an Object Type, Request Type, Report Type, Validation or Workflow.

Note

Special Commands can be used in an execution Workflow Step Source. After the Workflow Step Source is created (which contains the Special Commands) it can be dragged and dropped into a Workflow.

## Nesting Special Commands

Special Commands can be used within other Special Commands, but must be used within a command step. However, a Special Command cannot refer to itself.



# Chapter 4 Using Tokens

This chapter provides an overview of how to use Tokens in Kintana. This chapter discusses the following topics:

- *What are Tokens?*
- *Where Tokens Are Used*
- *Token Builder Window Overview*
- *Token Formats*
- *Token Evaluation*

## What are Tokens?

While configuring certain features in Kintana, it is often necessary to reference information that is undefined until the Kintana product is actually used a particular context. Instead of generating objects that are valid only in specific contexts, Kintana uses variables can be used to facilitate the creation of general objects that can be applied to a variety of contexts. These variables are called tokens.

There are two types of tokens found within Kintana: custom tokens and standard tokens. Standard tokens are provided with the product. Custom tokens are generated to suit specific needs. Each field of the following Kintana entities can be referenced as a custom token:

- Object Types
- Request Types and Request Header Types

- Report Types
- User Data
- Workflow Parameters

In addition, numerous standard Tokens are available that provide other useful pieces of information related to the Kintana system. For example, Kintana has a Token that represents the users currently logged onto the system.

## Where Tokens Are Used

Tokens can be used in many Kintana entity windows:

- Object Type commands
- Request Type commands
- Validation commands and SQL statements
- Report Type commands
- Executions and notifications for a Workflow
- Workflow Step commands
- Notifications in a Report Submission
- Special Command commands
- Notifications for Tasks

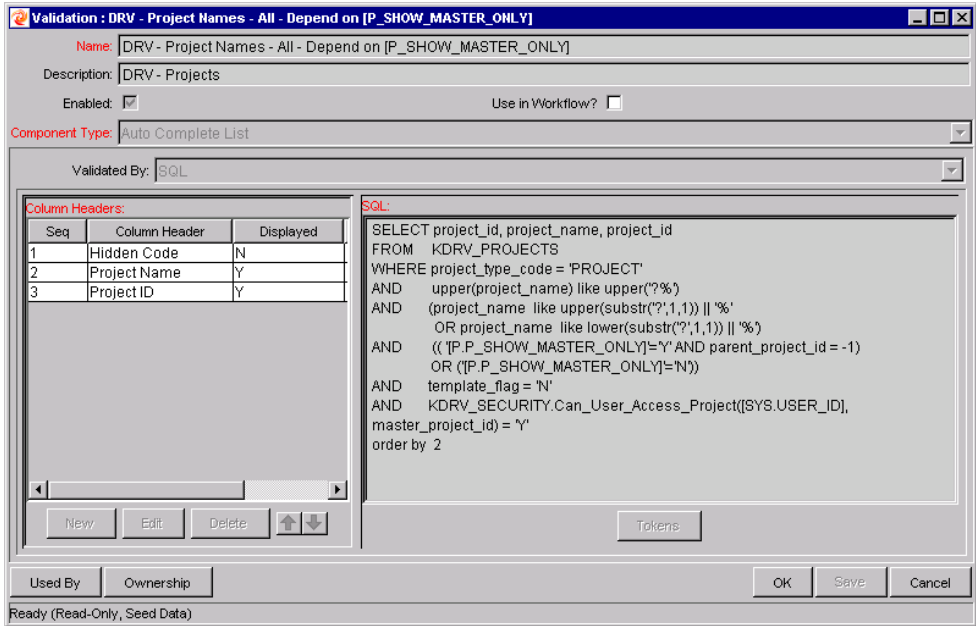


Figure 4-1 Example of a Token Used in a SQL Statement

# Token Builder Window Overview

In each of the entity windows listed in “Where Tokens Are Used” on page 42, you can create a token by opening the TOKEN BUILDER window.



Example

It is possible to open the TOKEN BUILDER window through the REQUEST TYPES window by doing the following:

1. Open a REQUEST TYPE WINDOW, either by generating a new Request Type or by opening an existing one.
2. Click the **COMMANDS** tab.
3. Click **NEW CMD**.
4. Click **TOKENS**. The TOKEN BUILDER window opens, as shown in *Figure 4-2*.
5. Use the TOKEN BUILDER window to help construct valid tokens.

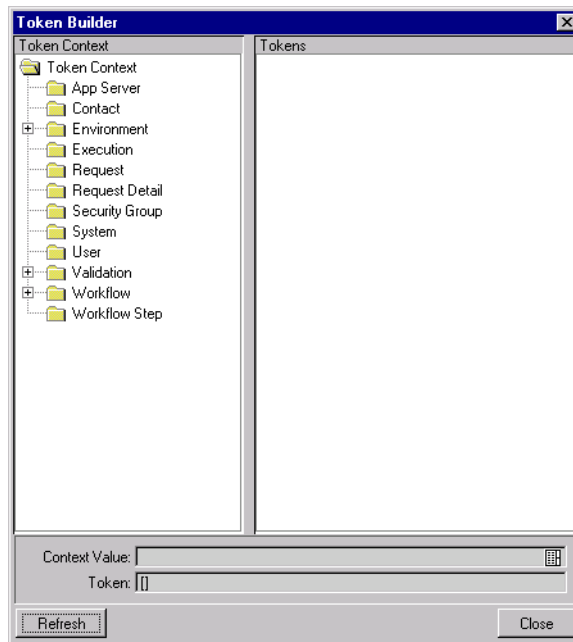


Figure 4-2 Token Builder Window

Folders are displayed in the left pane of the TOKEN BUILDER window. These folders contain groups of tokens that correspond to entities defined in the Kintana Product Suite. (See *“Tokens”* on page 95 for the entities and associated tokens.) For instance, the Packages folder contains tokens that reference various Package attributes. If the Packages folder is selected, the available Package tokens are displayed in the list in the right pane of the window.

Some entities (folders) have sub-entities (sub-folders) that can be referenced by tokens. Click the plus sign (+) next to an entity to see the list of sub-entities for an entity. Each sub-entity also has tokens, and it is possible to reference any of the tokens of sub-entities as well as tokens of the parent entity. For example, the Package Line entity is a sub-entity of the Package entity.

As entity folders and the subsequent tokens in the list are selected, a character string is constructed in the Token field at the bottom of the TOKEN BUILDER window. This is the formatted string used to reference the token. Either copy and paste the character string, or type this string where needed.

# Token Formats

Tokens can use one of several different formats, depending on how they are going to be evaluated. Tokens can be expressed in the following formats:

- *Default Format*
- *Explicit Entity Format*
- *User Data Format*
- *Parameter Format*
- *Sub-Entity Format*
- *Environment and Environment Application Tokens*  
 The Environment and Environment App entities evaluate differently than the other entities.

*Table 4-1* shows a list of entities and the formats each entity supports. Each format is discussed in a section following the table.

*Table 4-1. Entities*

| <b>Prefix (Entity)</b> | <b>Entity and Description</b>  | <b>User Data Format?</b> | <b>Parameter Format?</b> |
|------------------------|--|--------------------------|--------------------------|
| AS                     | App Server   | N                        | N                        |
| BGT                    | Budget   | Y                        | N                        |
| CON                    | Contact  | Y                        | N                        |
| DEST_ENV               | Destination Environment. If an App Code is specified, it will be used. Otherwise use only values from Env. | Y                        | N                        |
| DEST_ENV.APP           | Destination Environment (for the Environment Application). Only use App Code values, even if they're null. | Y                        | N                        |
| DEST_ENV.ENV           | Destination Environment. Ignores App Codes and only uses the ENV values.                                   | Y                        | N                        |
| DIST                   | Distribution   | Y                        | N                        |
| ENV                    | Environment  | Y                        | N                        |
| ENV.APP                | Environment (for the Environment Application). Only use App Code values, even if they're null.             | Y                        | N                        |
| ENV.ENV                | Environment. Ignores App Codes and only uses the ENV values.   | Y                        | N                        |
| EXEC                   | Execution  | N                        | N                        |
| NOTIF                  | Notification   | N                        | N                        |

Table 4-1. Entities

| Prefix (Entity) | Entity and Description  | User Data Format? | Parameter Format? |
|-----------------|---|-------------------|-------------------|
| ORG             | Organization Unit   | Y                 | N                 |
| PKG             | Package   | Y                 | N                 |
| PKG.PKGL        | Package (Package Line)  | Y                 | N                 |
| PKG.PEND        | Package (Pending Package)   | Y                 | N                 |
| PKGL            | Package Line  | Y                 | Y                 |
| PRG             | Program   | Y                 | N                 |
| PRJ             | Project   | Y                 | N                 |
| PRJD            | Project Details   | N                 | Y                 |
| REL             | Release   | N                 | N                 |
| REL.DIST        | Release (Distribution)  | Y                 | N                 |
| REQ             | Request   | Y                 | Y                 |
| REQ.PEND        | Request (Pending)   | N                 | N                 |
| REQD            | Request Details   | N                 | Y                 |
| RP              | Report Submission   | N                 | Y                 |
| RSCP            | Resource Pool   | Y                 | N                 |
| SG              | Security Group  | Y                 | N                 |
| SKL             | Skill   | Y                 | N                 |
| STFP            | Staffing Profile  | Y                 | N                 |
| SOURCE_ENV      | Source Environment  | Y                 | N                 |
| SOURCE_ENV.APP  | Source Environment (for Environment Application). Only use App Code values, even if they're null. | Y                 | N                 |
| SOURCE_ENV.ENV  | Source Environment. Ignores App Codes and only uses the ENV values.                               | Y                 | N                 |
| SYS             | System (Kintana)  | N                 | N                 |
| TSK             | Task  | Y                 | N                 |
| TSK.PEND        | Task (Pending)  | N                 | N                 |
| USR (User)      | User  | Y                 | N                 |
| VAL             | Validation  | N                 | N                 |
| VAL.VALUE       | Validation (Value). Use this format when you need to specify a specific Validation.               | Y                 | N                 |
| VALUE           | Validation (Value)  | Y                 | N                 |
| WF              | Workflow  | Y                 | N                 |
| WF.WFS          | Workflow (step). Use this format when you need to specify a specific Workflow.                    | N                 | Y                 |
| WFS             | Workflow Step   | Y                 | N                 |



## Default Format

Tokens are expressed as a prefix (a short name for the entity) followed by a token name. The prefix and token name are separated by a period and are enclosed in square brackets with no spaces:

[PREFIX.TOKEN\_NAME]



In the Kintana product, the token for the Package Number is expressed as:

[PKG.NUMBER]

The token for a Request's Workflow Name is expressed as:

[REQ.WORKFLOW\_NAME]

Certain Tokens also support a sub-format. This sub-format is required for certain entities in order to evaluate to the correct context. For example WF tokens will resolve to information related to the Workflow; whereas WF.WFS tokens will resolve to workflow step information. Token sub-formats are included in the prefix, appended to the parent prefix, separated by a period:

[PREFIX.SUB-PREFIX.TOKEN\_NAME]

Tokens are evaluated according to the current context of the Kintana Product Suite, which is derived based on information known at the time of evaluation. For more information, see [“Token Evaluation”](#) on page 58.

## Explicit Entity Format

It is possible to provide a specific context value for an entity. This allows the default context to be overridden. Some tokens can never be evaluated in the default context. In these cases, the context must be set using the explicit entity format, which is:

[PREFIX=“<entity name>”.TOKEN\_NAME]

The Token Builder helps generate Tokens of this format by providing a list of possible entity name values. When such a list is available, the Context Value

auto-complete field at the bottom of the Token Builder becomes enabled. Like any other auto-complete field, either type into the field to reduce the list or click the auto-complete icon in the field to open the Validate window. Once a value is selected, it is inserted into the token in the Token field, generating an explicit entity token (see *Figure 4-3*).

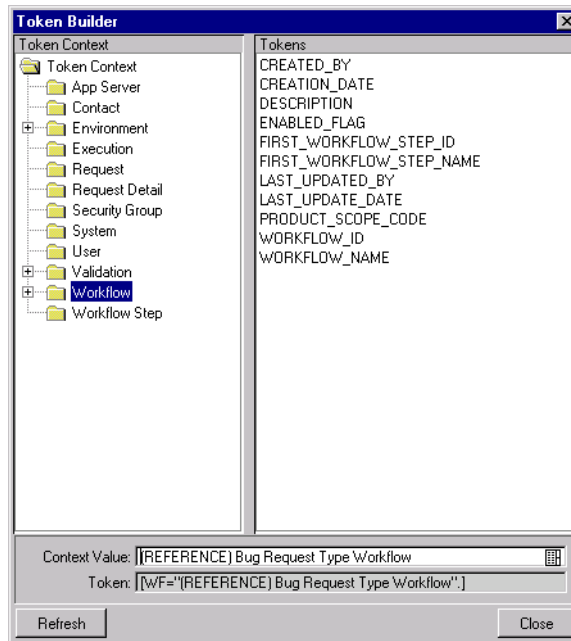


Figure 4-3 Explicit Entity Format



#### Example

Suppose the Email Address for the user “jsmith” is to be referenced. The token would look like this:

```
[USR="jsmith".EMAIL_ADDRESS]
```

To construct this token in the Token Builder window:

1. Select the USER folder. Available tokens are displayed in the list on the right pane. The CONTEXT VALUE field at the bottom of the Token Builder is enabled. The string [USR.] appears in the Token field below the Context Value field.
2. Click the auto-complete icon in the Context Value field. A Validate window opens with a list of users.
3. Scroll through the list to find user “jsmith.” Select this user and click **OK**. The string [USR="jsmith"] appears in the Token field.

4. In the list of tokens, select EMAIL\_ADDRESS. The string [USR="jsmith".EMAIL\_ADDRESS] appears in the Token field. This is the complete token. Since the token is now complete, the Token field becomes enabled.
5. Select the token
6. Select Ctrl+C on your keyboard to copy the token.
7. Select Ctrl+V on your keyboard to paste the token into another field.

### Using Tokens within Tokens

The explicit entity format can be used to put tokens within other tokens to get a value. For example, to print the description of the Workflow that is associated with Package #10203 the token would look like:

```
[WF="[PKG="10203".WORKFLOW_NAME]".DESCRIPTION]
```

This token would have to be built in two steps. First, build the Description token for the Workflow. Copy and paste that token into another field. Then build the Workflow Name token for the Package. Copy and paste that token within the Description token that was previously pasted.

Internally, this token is evaluated in two stages. The inner token is evaluated and the token has an internal representation of:

```
[WF="My_Workflow".DESCRIPTION]
```

The remaining token is evaluated and the final result is printed:

```
description of My_Workflow
```

*Table 4-2* includes a list of the Tokens that support the explicit entity format.



Note

It is important to note that <entity name> is case-sensitive and can contain spaces or other ASCII symbols.

Tokens for the User and Security Group entities can never be evaluated in the default format, and require the use of the explicit entity format. An example of this is the token [USR.EMAIL\_ADDRESS]. This token can never be evaluated because the Kintana Product Suite cannot determine to which one user it should refer.

Table 4-2. Tokens supporting explicit entity format

| Token Prefix | Example  | Acceptable Explicit Entry     |
|--------------|--|-------------------------------|
| BGT          | [BGT="Development Budget".CREATED_BY]            | Budget Name                   |
| CON          | [CON="Smith, John".PHONE_NUMBER]                 | Last Name, First Name         |
| ENV          | [ENV="KINTANA_SERVER".CLIENT_TRANSFERR_PROTOCOL] | Environment Name              |
| ORG          | [ORG="Project Managers".MANAGER_ID]              | Organization Unit Name        |
| PKG          | [PKG="30010".CREATED_BY]                         | Package Number                |
| REQ          | [REQ="30006".CREATED_BY]                         | Request Number                |
| RSCP         | [RSCP="Development Resources".CREATED_BY]        | Resource Pool Name            |
| SG           | [SG="Kintana Administrator".LAST_UPDATED_BY]     | Security Group Name           |
| SKL          | [SKL="Architect".AVERAGE_COST_RATE]              | Skill Name                    |
| STFP         | [STFP="Kintana Pilot".CREATED_BY]                | Staffing Profile Name         |
| USR          | [USR="jsmith".LAST_NAME]                         | User Name                     |
| VAL          | [VAL="Date".CREATED_BY]                          | Validation Name               |
| WF           | [WF="Dev -> Test -> Prod".CREATED_BY]            | Workflow Name                 |
| WF.WFS       | [WF="Workflow Name".WFS="1".STEP_NAME]           | Workflow Step Sequence Number |

## User Data Format

User Data fields use tokens differently, as shown below:

[PREFIX.UD.USER\_DATA\_TOKEN]

The Prefix is the name of the entity that has User Data. The modifier UD indicates that User Data for that entity is being referenced.

USER\_DATA\_TOKEN is the name of the token for the specific User Data field. For example, suppose that a field for Package User Data has been generated whose token is GAP\_NUMBER. In the default format, the token would look like:

[PKG.UD.GAP\_NUMBER]

where PKG indicates that the Package entity is being referenced, UD indicates that User Data is being referenced, and GAP\_NUMBER is the token name.

When User Data fields are generated, a Validation that has both a hidden and visible value can be used. For example, if the Validation ‘KNTA - Usernames - All’ is used, the hidden value is the User ID and the displayed value is the Username. The previous syntax references the hidden value only. To reference the visible value for a User Data field, the syntax shown below must be used:

[PREFIX.VUD.USER\_DATA\_TOKEN]

If the modifier VUD is used instead of UD, the visible User Data value is referenced.

Note

Drop Down Lists and Auto-complete Lists may have different hidden and displayed values. For all other Validations, the hidden and displayed values are identical.

When context can be determined, User Data tokens are displayed with the system-defined tokens in the Token Builder.

*Table 4-1* indicates which tokens support the user data format.

## Parameter Format

Object Type custom fields, Request Type custom fields, Request Header Type fields, Project fields, and Workflow Parameters use the Parameter format for tokens as shown below:

[PREFIX.P.PARAMETER\_TOKEN]

In this specific case, the Prefix is the name of the entity that uses a custom field. The modifier “P” indicates that Parameters for that entity are being referenced. PARAMETER\_TOKEN is the name of the token for the specific Parameter field.

Note

- Package Lines reference Object Type fields.
- Requests reference Request Type and Request Header Type fields.
- Workflows reference Workflow Parameters.

For example, suppose a field for an Object Type called GAP NUMBER (Token = GAP\_NUMBER) has been generated that is used on Package Lines. In the default format the token would look like:

[PKGL.P.GAP\_NUMBER]

where PKGL is the prefix since the Package Lines entity has been referenced, “P” indicates that Parameters have been referenced, and GAP\_NUMBER is the token name.

Custom fields store both a hidden and visible value. For example, if the field uses the Validation ‘KNTA - USERNAMES - ALL’, the hidden value is the USER ID and the displayed value is the USERNAME. The previous syntax references the hidden value only. To reference the visible value for a Parameter, use the syntax shown below:

[PREFIX.VP.PARAMETER\_TOKEN]

If the modifier ‘VP’ is used instead of ‘P’, the visible Parameter value is referenced.



Note

Drop Down Lists and Auto-complete Lists may have different hidden and displayed values. For all other Validations, the hidden and displayed values are identical.

## *Request Field Tokens*

Tokens can access information on custom fields included on a Request. These fields can be defined in a:

- Custom Request Type field
- Request Header Field (standard)
- Request Header Field (custom fields)
- Request Header Field (field groups)
- Table Component field

This section provides some additional details and examples on using Request Tokens.

### **Request Token Prefixes**

All fields defined in the Request Header Type (Field Group Fields, Custom Header Fields, and Standard Header Fields) use the REQ prefix. The following examples could use “P” or “VP.”

REQ.<standard header token>

Example: REQ.DEPARTMENT\_CODE

REQ.P.<custom header field token>

Example: REQ.P.BUSINESS\_UNIT

REQ.P.<field group token starting with KNTA\_>

Example: REQ.P.KNTA\_SKILL

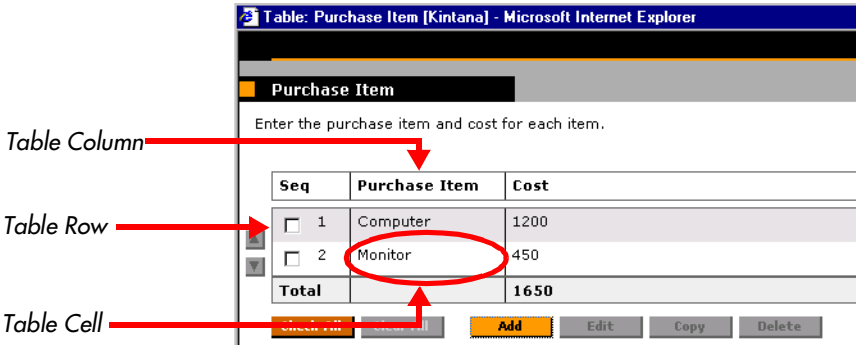
Fields defined in the Request Type use the REQD prefix. You can also access standard header fields using the REQD prefix.

REQD.P.<custom detail field>

REQD.<standard header token>

### Tokens in Request Table Components

When referring to items in a Table Component, the Tokens need to follow a specific formats. These formats differ depending on the item that you are referencing within the table. The following figure illustrates the basic elements of the table. These elements will be referenced when discussing the different options for referencing data within the table using Tokens.



The format [REQD.T.<TABLE\_TOKEN>] represents the table and specific tokens will be represented as [REQD.T.<TABLE\_TOKEN>.<SPECIFIC TOKENS>]. The following sections provide examples of the formats used for Tokens referencing items related to the table component.

For the examples, the following example will be used. A table component named EMPLOYEE with 4 columns: NAME OF EMPLOYEE, YEARS OF SERVICE of the

Employee, DEPARTMENT where the Employee belongs to, and the SALARY of the Employee. It is defined as follows.

|  |
|--|
| Table Component "Employee Table" with [EMPLOYEE] as the token.<br>Column 1 - Name of Employee; token = [NAME]<br>Column 2 - Years of Service; token = [YEARS_OF_SERVICE]<br>Column 3 - Department of Employee; token = [DEPARTMENT]<br>Column 4 - Salary of Employee; token = [SALARY] |
|--|

**To access the table row count from a Request context:**

[REQD.P.EMPLOYEE] - returns the raw row count without any descriptive information.

[REQD.VP.EMPLOYEE] - returns the row count with descriptive information. Example "13 Entry(s)".

WHERE: EMPLOYEE is the token given to a table component type.

**To access the Salary Column Total value from a Request context:**

[REQD.T.EMPLOYEE.TC.VP.SALARY.TOTAL]

WHERE: EMPLOYEE is the token given to a table component type and SALARY is the token name given the table's first column.

**To access the Name of the first employee in the table from a Request:**

[REQD.T.EMPLOYEE.TE="1".VP.NAME]

**To access the Code of the first employee in the table from a Request:**

[REQD.T.EMPLOYEE.TE="1".P.NAME]

**To access the Department Cell value of the current row. (Table Row Context)**

[TE.VP.DEPARTMENT]

You can use this Table Component Token in a Table Column Header Validation SQL or in a Table Component Rule SQL.



**To obtain a delimited list of a column's contents. (Request Context)**

```
[REQD.T.EMPLOYEE.TC.VP.NAME]
```

WHERE: EMPLOYEE is the token given to a table component type and SALARY is the token name given the table's first column.

This is particularly useful when a column is a list of user names and this list can be used for sending these users notification.

## Sub-Entity Format

Some entities have sub-entities that can be referenced. In the Token Builder, click the plus sign (+) next to an entity to see the list of its sub-entities. To reference a token from a sub-entity, in the context of a parent entity, use the syntax shown below:

```
[PREFIX.SUB_ENTITY_PREFIX.TOKEN]
```

In this case, the Prefix is the name of the entity, the sub-entity Prefix is the prefix for a sub-entity, and Token is a token of the sub-entity. Most of the time, it is not necessary to use this syntax. However, it is possible to reference specific sub-entities using the explicit entity syntax.

For example, to reference the Step Name of the Workflow Step in the current context, both of the following tokens mean the same thing:

```
[WFS.STEP_NAME]
```

```
[WF.WFS.STEP_NAME]
```

However, to reference the Step Name of the first Workflow Step for the current Workflow, use the token:

```
[WF.WFS="1".STEP_NAME]
```

By not using the explicit entity format for the Workflow entity, the token indicates to use the Workflow in the current context. But by using the explicit entity format for the Workflow Step entity, the current context is overridden and a specific Workflow Step is referenced. In contrast, to reference the Step Name of the first Workflow Step on a Workflow whose name is 'my workflow', use the token:

```
[WF="my workflow".WFS="1".STEP_NAME]
```

With this token, the current context for both the Workflow and the Workflow Step is overridden.

## Environment and Environment Application Tokens

Tokens for the Environments and Environment Application entities can have many different forms depending on the information to be referenced. During Object Type command execution, there is generally a source and a destination Environment. The token prefixes SOURCE\_ENV and DEST\_ENV are used to reference the current source and destination, respectively. For example:

```
[SOURCE_ENV.DB_USERNAME]
```

```
[DEST_ENV.SERVER_BASE_PATH]
```

In addition, a general ENV Prefix can be used in the explicit entity format to reference specific Environments. For example:

```
[ENV="Prod".CLIENT_USERNAME]
```

During normal environment token evaluation, the evaluation engine first looks at the App Code on the Package Line (if one is specified). If the corresponding App Code token has a value, then the value is used. Otherwise, if no App Code was specified or the App Code token has no value, the corresponding base Environment information is used.

To override the normal Environment token evaluation and look only at the Environment information (without first checking for the App Code), construct the SOURCE\_ENV and DEST\_ENV tokens as shown in the following examples:

```
[SOURCE_ENV.ENV.DB_USERNAME]
```

```
[DEST_ENV.ENV.SERVER_BASE_PATH]
```

```
[ENV="Prod".ENV.CLIENT_USERNAME]
```

The evaluation engine can be instructed to look only at the App Code information (without checking the base Environment information if the App Code token has no value). Construct the SOURCE\_ENV and DEST\_ENV tokens as follows:

```
[SOURCE_ENV.APP.DB_USERNAME]
```

```
[DEST_ENV.APP.SERVER_BASE_PATH]
```

[ENV="Prod".APP.CLIENT\_USERNAME]

The prefix 'APP' can only be used in the sub-entity format. For example, the following token is invalid, since a context Environment that includes the app code has not been specified.

[APP.SERVER\_BASE\_PATH]

In addition, the explicit entity format can be used with the App Code entity to reference a specific App Code, as shown in these examples:

[SOURCE\_ENV.APP="AR".DB\_USERNAME]

[DEST\_ENV.APP="OE".SERVER\_BASE\_PATH]

[ENV="Prod".APP="HR".CLIENT\_USERNAME]

For example, suppose objects are being migrated on a Package Line at a given Workflow Step, and the line uses App Code "HR". The Workflow Step has 'QA' as the Source Environment and 'Prod' as the Destination Environment. Table C-2 shows other attributes of the Environments and Applications.

*Table 4-3. Sample Environment and App Attributes*

| Environment | App Code | Server Base Path |
|-------------|----------|------------------|
| QA          |          | /qa              |
| QA          | OE       | /qa/oe           |
| QA          | HR       | /qa/hr           |
| Prod        |          | /prod            |
| Prod        | OE       | /prod/oe         |
| Prod        | HR       | <no value>       |

Given this setup, Table C-3 shows some sample tokens and how each would evaluate.

*Table 4-4. Sample Environment Tokens*

| Token                             | Evaluation |
|-----------------------------------|------------|
| [SOURCE_ENV.SERVER_BASE_PATH]     | /qa/hr     |
| [DEST_ENV.SERVER_BASE_PATH]       | /prod      |
| [SOURCE_ENV.ENV.SERVER_BASE_PATH] | /qa        |
| [DEST_ENV.ENV.SERVER_BASE_PATH]   | /prod      |
| [SOURCE_ENV.APP.SERVER_BASE_PATH] | /qa/hr     |
| [DEST_ENV.APP.SERVER_BASE_PATH]   | <no value> |

|                                      |        |
|--------------------------------------|--------|
| [ENV="QA".APP="OE".SERVER_BASE_PATH] | /qa/oe |
|--------------------------------------|--------|

## Token Evaluation

Tokens are evaluated at the point when Kintana must know their context-specific values. At the time of evaluation, the token evaluation engine gathers information from the current context and tries to derive the value for the token. Values can only be derived for specific, known contexts. (The current context is defined as the current Package, Package Line, Request, Project, Workflow Step, Source and Destination Environments, and so on.)

The token evaluation engine takes as many passes as necessary to evaluate all of the tokens, so one token can be nested within another token. During each pass, if the evaluation engine finds a valid token, it replaces that token with its derived value. Tokens that are invalid for any reason (such as the token is misspelled or no context is available) are left alone.

For example, suppose an Object Type Command has the following Bourne-shell script segment as one of its Command Steps:

```
if [ ! -f [PKGL.P.P_SUB_PATH]/[PKGL.P.P_BASE_FILENAME].fmx
]; then exit 1; fi
```

At the time of execution [PKGL.P.P\_SUB\_PATH] = "Forms" and [PKGL.P.P\_BASE\_FILENAME] = "obj\_maint". After token evaluation, this Command step would reduce to:

```
if [ ! -f Forms/obj_maint.fmx ]; then exit 1; fi
```

As another example, suppose a User Data field has been generated for all Users called 'MANAGER.' The email address of the manager of the person who generated a Request could be found using the token:

```
[USR=" [USR=" [REQ.CREATED_BY_NAME] ".VUD.MANAGER] ".EMAIL_ADDRESS]
```

The token evaluation engine would first evaluate the innermost token ([REQ.CREATED\_BY\_NAME]). Once that is complete, the next token ([USR="<name>".VUD.MANAGER]) is evaluated. Finally, the outermost token is evaluated, giving the manager's email address.

Tokens are evaluated at different points based on the type of token. Tokens used in Object Type Parameters and Commands are evaluated during Command execution. Tokens in a Validation SQL statement are evaluated just

before that statement is executed (such as generating a new Package Line).  
Tokens in an email Notification are evaluated when a Notification is generated.



## Appendix



# System Special Commands

Special Commands are commands with variable parameters used in Object Type, Report Type, Request Type and Workflow command steps to perform a variety of functions such as copying files between environments and establishing connections to environments for remote command execution. The Kintana Product Suite includes two types of Special Commands:

1. Predefined Special Commands - These commands are read-only and are denoted by the naming convention 'ksc\_command\_name'
2. User Defined Special Commands - These commands are user-defined and are denoted by the naming convention 'sc\_command\_name'.

This appendix discusses Kintana's pre-defined Special Commands:

- [“Special Commands in Kintana”](#) on page 61
- [“Summary of All Special Command Parameters”](#) on page 89

## Special Commands in Kintana

The following sections describe each Special Command in detail:

- [“ksc\\_connect Special Commands”](#) on page 62
- [“ksc\\_exit”](#) on page 66
- [“ksc\\_copy Special Commands”](#) on page 66
- [“ksc\\_respond”](#) on page 73
- [“ksc\\_simple\\_respond”](#) on page 73
- [“ksc\\_local\\_exec”](#) on page 75

- “*ksc\_replace*” on page 76
- “*ksc\_set*” on page 76
- “*ksc\_set\_env*” on page 77
- “*ksc\_store*” on page 78
- “*ksc\_comment*” on page 79
- “*ksc\_concsub*” on page 79
- “*ksc\_begin\_script / ksc\_end\_script*” on page 80
- “*ksc\_copy\_script Special Commands*” on page 82
- “*ksc\_om\_migrate*” on page 84
- “*ksc\_capture\_output*” on page 85
- “*ksc\_gl\_migrate*” on page 86
- “*ksc\_parse\_jcl*” on page 87
- “*ksc\_submit\_job*” on page 87
- “*ksc\_set\_exit\_value*” on page 87
- “*ksc\_clear\_exit\_value*” on page 88
- “*ksc\_run\_sql*” on page 88

## **ksc\_connect Special Commands**

The `ksc_connect` Special Commands instruct the execution engine to open a connection to a specified environment. This command initiates a TELNET,SSH or SSH2 session with the server or client defined for the environment. It then sends all of the command steps that follow it directly to the machine as though someone was actually typing the command on that machine. In this way, the execution engine is able to run virtually any command-line directive that the machine understands.



Note

All `ksc_connect` Special Commands must end with the ‘`ksc_exit`’ Special Command to exit the TELNET, SSH or SSH2 session.



*ksc\_connect\_dest\_client*

This command initiates a TELNET, SSH or SSH2 session with the client of the destination environment. The destination environment refers to the destination environment of the Workflow Step initiating command execution.

Table A-1. *ksc\_connect\_dest\_client* Parameters

| Parameter               | Default Token                             | Description   |
|-------------------------|---|---|
| USERNAME                | [DEST_ENV.CLIENT_USER<br>NAME]            | Username on [DEST_ENV].   |
| PASSWORD                | [DEST_ENV.CLIENT_PASS<br>WORD]            | Password on [DEST_ENV].   |
| NT_DOMAIN               | [DEST_ENV.CLIENT_NT_D<br>OMAIN]           | Windows NT Domain name of [DEST_ENV].   |
| DEST_BASE_PA<br>TH      | [DEST_ENV.CLIENT_BASE<br>_PATH]           | Base Path of [DEST_ENV].  |
| CONNECTION_<br>PROTOCOL | [DEST_ENV.CLIENT_CON<br>PROTOCOL_MEANING] | Specifies the connection protocol. Possible values are listed in Validation "CONNECTION_PROTOCOL".                  |
| DEST_ENV                | [DEST_ENV]                                | Name of the destination environment to be used instead of the destination environment on the current Workflow Step. |

**Example Using *ksc\_connect\_dest\_client***

```
# Make a remote connection to the client of the
# destination environment defined for the current
# workflow step.

ksc_connect_dest_client
<commands>
ksc_exit

# Make a remote connection to the client defined for
# the environment named 'STAGING'.

ksc_connect_dest_client DEST_ENV="STAGING"
<commands>
ksc_exit
```

*ksc\_connect\_dest\_server*

This command initiates a TELNET, SSH or SSH2 session with the server of the destination environment. The destination environment refers to the destination environment of the Workflow Step initiating command execution.

Table A-2. *ksc\_connect\_dest\_server* Parameters

| Parameter           | Default Token                                 | Description   |
|---------------------|---|---|
| USERNAME            | [DEST_ENV.SERVER_USERNAME]                    | Username on [DEST_ENV].   |
| PASSWORD            | [DEST_ENV.SERVER_PASSWORD]                    | Password on [DEST_ENV].   |
| NT_DOMAIN           | [DEST_ENV.SERVER_NT_DOMAIN]                   | Windows NT Domain name of [DEST_ENV].   |
| DEST_BASE_PATH      | [DEST_ENV.SERVER_BASE_PATH]                   | Base Path of [DEST_ENV].  |
| CONNECTION_PROTOCOL | [DEST_ENV.SERVER_CONNECTION_PROTOCOL_MEANING] | Specifies the connection protocol. Possible values are listed in Validation "CONNECTION_PROTOCOL".                  |
| DEST_ENV            | [DEST_ENV.ENVIRONMENT_NAME]                   | Name of the destination environment to be used instead of the destination environment on the current Workflow Step. |

### Example using *ksc\_connect\_dest\_server*

```
# Make a remote connection to the server of the
# destination environment defined for the current
# workflow step.

ksc_connect_dest_server
<commands>
ksc_exit

# Make a remote connection to the server defined for
# the environment named 'Staging'.

ksc_connect_dest_server DEST_ENV="STAGING"
<commands>
ksc_exit
```

### *ksc\_connect\_source\_client*

This command initiates a TELNET, SSH or SSH2 session with the client of the source environment. The source environment refers to the source environment of the Workflow Step initiating command execution.

Table A-3. *ksc\_connect\_source\_client* Parameters

| Parameter | Default Token                | Description               |
|-----------|------------------------------|---------------------------|
| USERNAME  | [SOURCE_ENV.CLIENT_USERNAME] | Username on [SOURCE_ENV]. |

Table A-3. *ksc\_connect\_source\_client* Parameters

| Parameter           | Default Token                                   | Description   |
|---------------------|---|---|
| PASSWORD            | [SOURCE_ENV.CLIENT_PASSWORD]                    | Password on [SOURCE_ENV].   |
| NT_DOMAIN           | [SOURCE_ENV.CLIENT_NT_DOMAIN]                   | Windows NT Domain name of [SOURCE_ENV].   |
| SOURCE_BASE_PATH    | [SOURCE_ENV.CLIENT_BASE_PATH]                   | Base Path of [SOURCE_ENV].  |
| CONNECTION_PROTOCOL | [SOURCE_ENV.CLIENT_CONNECTION_PROTOCOL_MEANING] | Specifies the connection protocol. Possible values are listed in Validation "CONNECTION_PROTOCOL".        |
| SOURCE_ENV          | [SOURCE_ENV]                                    | Name of the source environment to be used instead of the source environment on the current Workflow Step. |

### Example using *ksc\_connect\_source\_client*

```
# Make a remote connection to the client of the source
# environment defined for the current workflow step.

ksc_connect_source_client
<commands>
ksc_exit

# Make a remote connection to the client defined for
# the environment named 'STAGING'.

ksc_connect_source_client SOURCE_ENV="STAGING"
<commands>
ksc_exit
```

### *ksc\_connect\_source\_server*

This command initiates a TELNET, SSH or SSH2 session with the server of the source environment. The source environment refers to the source environment of the Workflow Step initiating command execution.

Table A-4. *ksc\_connect\_source\_server* Parameters

| Parameter | Default Token                | Description               |
|-----------|------------------------------|---------------------------|
| USERNAME  | [SOURCE_ENV.SERVER_USERNAME] | Username on [SOURCE_ENV]. |
| PASSWORD  | [SOURCE_ENV.SERVER_PASSWORD] | Password on [SOURCE_ENV]. |

Table A-4. *ksc\_connect\_source\_server* Parameters

| Parameter           | Default Token                                   | Description   |
|---------------------|---|---|
| NT_DOMAIN           | [SOURCE_ENV.SERVER_NT_DOMAIN]                   | Windows NT Domain name of [SOURCE_ENV].   |
| SOURCE_BASE_PATH    | [SOURCE_ENV.SERVER_BASE_PATH]                   | Base Path of [SOURCE_ENV].  |
| CONNECTION_PROTOCOL | [SOURCE_ENV.SERVER_CONNECTION_PROTOCOL_MEANING] | Specifies the connection protocol. Possible values are listed in Validation "CONNECTION_PROTOCOL".        |
| SOURCE_ENV          | [SOURCE_ENV]                                    | Name of the source environment to be used instead of the source environment on the current Workflow Step. |

### Examples using *ksc\_connect\_source\_server*

```
# Make a remote connection to the server of the source
# environment defined for the current workflow step.
```

```
ksc_connect_source_server
<commands>
ksc_exit
```

```
# Make a remote connection to the server defined for
# the environment named 'STAGING'.
```

```
ksc_connect_source_server SOURCE_ENV="STAGING"
<commands>
ksc_exit
```

## ksc\_exit

This command exits the TELNET, SSH or SSH2 session initiated by the *ksc\_connect Special Commands* described above.

For examples using *ksc\_exit*, please see the examples in section "*ksc\_connect Special Commands*" on page 62.

## ksc\_copy Special Commands

The *ksc\_copy* Special Commands provide the mechanism for transferring files to and from the various environments defined in the Kintana Product Suite.

Note that the default use of these commands requires that the entity containing the command has three fields with the following tokens defined:

- 1 [P.P\_FILENAME]

- 2 [P.P\_FILE\_TYPE]
- 3 [P.P\_SUB\_PATH]

If they are not defined as a part of the entity, they must be passed as parameters or the command will fail. Please see the examples below.



Files are copied using either FTP, SCP or SCP2, depending on the configuration of the environment.

*ksc\_copy\_client\_client*

This command copies a file from the source client environment to the destination client environment.

Table A-5. *ksc\_copy\_client\_client* Parameters

| Parameter        | Default Token                 | Description  |
|------------------|-------------------------------|--|
| SUB_PATH         | [P.P_SUB_PATH]                | The sub-directory that should be used to locate the file relative to the base path of each environment.                            |
| SOURCE_BASE_PATH | [SOURCE_ENV.CLIENT_BASE_PATH] | The base path of the source client environment to be used instead of what is defined for the current source environment.           |
| DEST_BASE_PATH   | [DEST_ENV.CLIENT_BASE_PATH]   | The base path of the destination client environment to be used instead of what is defined for the current destination environment. |
| FILENAME         | [P.P_FILENAME]                | Name of the file to be copied.   |
| FILE_TYPE        | [P.P_FILE_TYPE]               | The file type associated with the file (ASCII or BINARY).  |
| SOURCE_ENV       | [SOURCE_ENV]                  | Name of the source environment to be used instead of the source environment on the current Workflow Step.                          |
| DEST_ENV         | [DEST_ENV]                    | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

**Example #1 using *ksc\_copy\_client\_client***

```
# Copy a file between source and destination clients.
ksc_copy_client_client SUB_PATH="forms"
```

```

FILENAME="[P.P_MODULE].fmb" FILE_TYPE="BINARY"

# Copy a file between the client defined in the 'STAGING'
# environment and the destination client.

ksc_copy_client_client DEST_ENV="STAGING"

```

### Example #2 using ksc\_copy\_client\_client

```

# Override the base path of the destination directory.

ksc_copy_client_client DEST_BASE_PATH="/u1/datatree/ex1"
SUB_PATH="." FILENAME="[P.P_MODULE].fmb"
FILE_TYPE="BINARY"

```

### *ksc\_copy\_client\_server*

This command copies a file from the source client environment to the destination server environment.

Table A-6. *ksc\_copy\_client\_server* Parameters

| Parameter        | Default Token                 | Description  |
|------------------|-------------------------------|--|
| SUB_PATH         | [P.P_SUB_PATH]                | The sub-directory that should be used to locate the file relative to the base path of each environment.                            |
| SOURCE_BASE_PATH | [SOURCE_ENV.CLIENT_BASE_PATH] | The base path of the source client environment to be used instead of what is defined for the current source environment.           |
| DEST_BASE_PATH   | [DEST_ENV.SERVER_BASE_PATH]   | The base path of the destination server environment to be used instead of what is defined for the current destination environment. |
| FILENAME         | [P.P_FILENAME]                | Name of the file to be copied.   |
| FILE_TYPE        | [P.P_FILE_TYPE]               | The file type associated with the file (ASCII or BINARY).  |
| SOURCE_ENV       | [SOURCE_ENV]                  | Name of the source environment to be used instead of the source environment on the current Workflow Step.                          |
| DEST_ENV         | [DEST_ENV]                    | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

### Example using ksc\_copy\_client\_server

```

# Copy a file between source client and
# destination server.

```

```
ksc_copy_client_server SUB_PATH="install/sql"
FILENAME=" [P.P_SQL_SCRIPT] " FILE_TYPE="ASCII"
```

### *ksc\_copy\_server\_client*

This command copies a file from the source server environment to the destination client environment.

Table A-7. *ksc\_copy\_server\_client* Parameters

| Parameter        | Default Token                     | Description  |
|------------------|-----------------------------------|--|
| SUB_PATH         | [P.P_SUB_PATH]                    | The sub-directory that should be used to locate the file relative to the base path of each environment.                            |
| SOURCE_BASE_PATH | [SOURCE_ENV.<br>SERVER_BASE_PATH] | The base path of the source server environment to be used instead of what is defined for the current source environment.           |
| DEST_BASE_PATH   | [DEST_ENV.<br>CLIENT_BASE_PATH]   | The base path of the destination client environment to be used instead of what is defined for the current destination environment. |
| FILENAME         | [P.P_FILENAME]                    | Name of the file to be copied.   |
| FILE_TYPE        | [P.P_FILE_TYPE]                   | The file type associated with the file (ASCII or BINARY).  |
| SOURCE_ENV       | [SOURCE_ENV]                      | Name of the source environment to be used instead of the source environment on the current Workflow Step.                          |
| DEST_ENV         | [DEST_ENV]                        | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

### Example using *ksc\_copy\_server\_client*

```
# Copy a file between source server and
# destination client.

ksc_copy_server_client SUB_PATH=" [P.P_SUB_DIRECTORY] "
FILE_TYPE=" [P.P_FILE_TYPE] "
```

*ksc\_copy\_server\_server*

This command copies a file from the source server environment to the destination server environment.

Table A-8. *ksc\_copy\_server\_server* Parameters

| Parameter        | Default Token                     | Description  |
|------------------|-----------------------------------|--|
| SUB_PATH         | [P.P_SUB_PATH]                    | The sub-directory that should be used to locate the file relative to the base path of each environment.                            |
| SOURCE_BASE_PATH | [SOURCE_ENV.<br>SERVER_BASE_PATH] | The base path of the source server environment to be used instead of what is defined for the current source environment.           |
| DEST_BASE_PATH   | [DEST_ENV.<br>SERVER_BASE_PATH]   | The base path of the destination server environment to be used instead of what is defined for the current destination environment. |
| FILENAME         | [P.P_FILENAME]                    | Name of the file to be copied.   |
| FILE_TYPE        | [P.P_FILE_TYPE]                   | The file type associated with the file (ASCII or BINARY).  |
| SOURCE_ENV       | [SOURCE_ENV]                      | Name of the source environment to be used instead of the source environment on the current Workflow Step.                          |
| DEST_ENV         | [DEST_ENV]                        | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

### Example using *ksc\_copy\_server\_server*

```
# Copy a file between source and destination servers.
ksc_copy_server_server FILENAME="[P.P_FILE]"

# Copy a file between the source server and the
# destination server overriding the base bath.

ksc_copy_server_server FILENAME="install_driver.sh"
DEST_BASE_PATH="/u2/app/drivers"

# Copy a form between the 'STAGING' and destination servers.

ksc_copy_server_server SOURCE_ENV="STAGING" SUB_PATH="forms"
FILENAME="[P.P_MODULE].fmb" FILE_TYPE="BINARY"
```



*ksc\_copy\_client\_tmp*

This command copies a file from the source client environment to the temporary Package transfer directory on the application server. This temporary directory is automatically cleaned up after an execution completes and can be referenced using the [AS.PKG\_TRANSFER\_PATH] token.

Table A-9. *ksc\_copy\_server\_tmp* Parameters

| Parameter        | Default Token                 | Description  |
|------------------|-------------------------------|--|
| SUB_PATH         | [P.P_SUB_PATH]                | The sub-directory that should be used to locate the file relative to the base path of each environment.                  |
| SOURCE_BASE_PATH | [SOURCE_ENV.CLIENT_BASE_PATH] | The base path of the source client environment to be used instead of what is defined for the current source environment. |
| FILENAME         | [P.P_FILENAME]                | Name of the file to be copied.   |
| FILE_TYPE        | [P.P_FILE_TYPE]               | The file type associated with the file (ASCII or BINARY).  |
| SOURCE_ENV       | [SOURCE_ENV]                  | Name of the source environment to be used instead of the source environment on the current Workflow Step.                |

*ksc\_copy\_server\_tmp*

This command copies a file from the source server environment to the temporary Package transfer directory on the application server. This temporary directory is automatically cleaned up after an execution completes and can be referenced using the [AS.PKG\_TRANSFER\_PATH] token.

Table A-10. *ksc\_copy\_server\_tmp* Parameters

| Parameter        | Default Token                 | Description  |
|------------------|-------------------------------|--|
| SUB_PATH         | [P.P_SUB_PATH]                | The sub-directory that should be used to locate the file relative to the base path of each environment.                  |
| SOURCE_BASE_PATH | [SOURCE_ENV.SERVER_BASE_PATH] | The base path of the source server environment to be used instead of what is defined for the current source environment. |
| FILENAME         | [P.P_FILENAME]                | Name of the file to be copied.   |
| FILE_TYPE        | [P.P_FILE_TYPE]               | The file type associated with the file (ASCII or BINARY).  |

Table A-10. *ksc\_copy\_server\_tmp* Parameters

| Parameter  | Default Token | Description   |
|------------|---------------|---|
| SOURCE_ENV | [SOURCE_ENV]  | Name of the source environment to be used instead of the source environment on the current Workflow Step. |

*ksc\_copy\_tmp\_client*

This command copies a file from the temporary Package transfer directory on the application server to the destination client environment. This temporary directory is automatically cleaned up after an execution completes and can be referenced using the [AS.PKG\_TRANSFER\_PATH] token.

Table A-11. *ksc\_copy\_server\_tmp* Parameters

| Parameter      | Default Token               | Description   |
|----------------|-----------------------------|---|
| SUB_PATH       | [P.P_SUB_PATH]              | The sub-directory that should be used to locate the file relative to the base path of each environment.                     |
| DEST_BASE_PATH | [DEST_ENV.CLIENT_BASE_PATH] | The base path of the dest server environment to be used instead of what is defined for the current destination environment. |
| FILENAME       | [P.P_FILENAME]              | Name of the file to be copied.  |
| FILE_TYPE      | [P.P_FILE_TYPE]             | The file type associated with the file (ASCII or BINARY).   |
| DEST_ENV       | [DEST_ENV]                  | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.         |

*ksc\_copy\_tmp\_server*

This command copies a file from the temporary Package transfer directory on the application server to the destination server environment. This temporary directory is automatically cleaned up after an execution completes and can be referenced using the [AS.PKG\_TRANSFER\_PATH] token.

Table A-12. *ksc\_copy\_server\_tmp* Parameters

| Parameter      | Default Token                   | Description  |
|----------------|---------------------------------|--|
| SUB_PATH       | [P.P_SUB_PATH]                  | The sub-directory that should be used to locate the file relative to the base path of each environment.                            |
| DEST_BASE_PATH | [DEST_ENV.<br>SERVER_BASE_PATH] | The base path of the destination server environment to be used instead of what is defined for the current destination environment. |
| FILENAME       | [P.P_FILENAME]                  | Name of the file to be copied.   |
| FILE_TYPE      | [P.P_FILE_TYPE]                 | The file type associated with the file (ASCII or BINARY).  |
| DEST_ENV       | [DEST_ENV]                      | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

## ksc\_respond

This command is currently only used to support Kintana's Patch\*Applicator. This command is able to intelligently respond to interactive prompts generated by the Oracle 'adpatch' program. General use of this Special Command for arbitrary programs is not yet supported. For simple interactive programs, see "[ksc\\_simple\\_respond](#)" on page 73.

## ksc\_simple\_respond

This command executes an interactive UNIX command on a remote computer. This command is useful when the command to be executed will prompt for additional information (such as the UNIX 'su' command to switch user accounts) or may not return an exit code upon completion (such as starting up a new shell using 'sh').



Note

This command can only be used from within a remote execution session, i.e. between 'ksc\_connect' and 'ksc\_exit' commands.

The following syntax is supported:

```
ksc_simple_respond "command"
ksc_simple_respond "command" "prompt 1" "response 1"
["prompt 2" "response 2" ... ]
```

```
ksc_simple_respond "command" -hide "prompt 1" "response 1"  
["prompt 2" "response 2" ... ]
```

There can be as many prompt-response pairs as necessary. Each prompt must be matched with a response, even if the response is an empty string. The prompts must appear in the exact order they will be displayed as the command is run. All arguments must be enclosed in quotes. In addition, if the command or any of the arguments contains double quotes (“), any other character can be used as the quote character. The first character after the string ‘ksc\_simple\_respond’ will be interpreted as the quote character, and that character must appear at the beginning and end of each argument.

By using the -hide option, the value passed in for the response will not be displayed in the execution log. In the log, the value will be displayed as \*\*\*\*\*. This flag should be used for each prompt/response pair that needs this treatment.



Note

The Kintana execution engine will wait for each specified prompt. If a prompt does not appear for some reason, then the execution engine will continue to wait for it until the command times out.

### *Examples using ksc\_simple\_respond*

If it becomes necessary to invoke a new shell while in a remote session, it would be ideal to just use the command ‘sh’. However, this can cause the Kintana execution engine to wait indefinitely while waiting for an exit code. To avoid this problem, the ‘sh’ command can be encapsulated in a ksc\_simple\_respond command with no prompts, as follows:

```
ksc_simple_respond "sh"
```

As another example, suppose it becomes necessary to switch to another user account while in a remote session using the ‘su’ command. This command always prompts for password, unless performed by a root user. By utilizing the -hide feature, the password will not be displayed in the execution logs. This interactivity can be handled using ksc\_simple\_respond as follows:

```
ksc_simple_respond "su <username>" -hide "word:"  
"<password>"
```

Note that “word:” was used as the prompt instead of the entire word “password:”. The Kintana execution engine will wait for the specified prompt string, whether it is all, or just a part, of the prompt text.

As one more example, consider the following Bourne shell command:

```
echo "Enter a string:\c"; read str; echo $str
```

Normally, this command line would cause the Kintana execution engine to hang while waiting for an exit code (the command will never exit because it's waiting for input), which would eventually timeout when the execution timeout time is reached. Use `ksc_simple_respond` to process this command as follows (note it should be entered on a single line):

```
ksc_simple_respond #echo "Enter a string:\c"; read str; echo
  $str# #a string:# #my_value#
```

Since the command line contained double quotes, we chose to use the pound sign (#) as the quote character. During execution, this command step would prompt “Enter a string:” and wait for some input. The string “my\_value” would be entered automatically, this value would be echoed to the output device (in this case, the execution log), and execution would continue as normal with the next command step.

## **ksc\_local\_exec**

This command invokes a local process on the machine running the Kintana application server. It can be used to run any program that does not require interactive input. Each call using ‘`ksc_local_exec`’ is an independent process. It does not execute in the context of other commands that precede it. The starting directory for the processes generated using ‘`ksc_local_exec`’ is the home directory of the Kintana server. Full paths to the executable being called are necessary if the Kintana server does not have the correct system path information.



Note

The `ksc_local_exec` command does not open a TELNET, SSH or SSH2 connection to the Kintana server. It operates by creating a new child process on the machine that is running the Kintana application server. Therefore, the user account and password for this process will be the same as the account and password used to start the Kintana application server.

### *Example using `ksc_local_exec`*

```
# Rename existing file 'file.txt' to 'newfile.txt'
ksc_local_exec mv file.txt newfile.txt
```

```
# Run a DOS batch file
ksc_local_exec cmd /c runme.bat
```

Kintana system commands do not invoke either Unix shells or DOS shells. This means that the following code segment using ‘`ksc_local_exec`’ is not valid because it cannot use the ‘pipe’ (|) or redirect commands (>):

```
ksc_local_exec cat names.txt | grep address > file.out
```

An effective way to use the `ksc_local_exec` command is to put a series of commands into a `.sh` file, and then execute the `.sh` file. An example is shown below:

```
ksc_begin_script + [AS.CR_TRANSFER_PATH] run.sh
..
<series of commands>
ksc_end_script

ksc_local_exe ksh run.sh
```

## ksc\_replace

This command is used to edit the contents of a file and place it into another file. It works in a way similar to the ‘sed’ utility and supports the same substituting expressions.

The files must be located on the Kintana server in the `[AS.PKG_TRANSFER_PATH]` directory. This requires the use of the `ksc_copy_tmp_*` commands.

Table A-13. *ksc\_replace* Parameters

| Parameter | Default Token  | Description  |
|-----------|----------------|--|
| FILENAME  | [P.P_FILENAME] | Name of the source file to be edited.                                |
| OUTFILE   | [OUTFILE]      | Name of the output file after applying the substitution expressions. |
| SUBST     |                | The substitution expression.   |

### Example using *ksc\_replace*

```
ksc_copy_server_tmp FILENAME="config.template"
FILE_TYPE="ASCII"

ksc_replace FILENAME="config.template" OUTFILE="config.cfg"
SUBST="s/NAME/[P.NAME]/g"

ksc_copy_tmp_server FILENAME="config.cfg"
```

## ksc\_set

This command sets the value of a temporary variable which may be used to manage command conditions or aid in command processing.

The following syntax is supported:

```
ksc_set VARIABLE="Value"
```

To reference the value of this variable, use the familiar token syntax without any prefix. Unlike the 'ksc\_store' command, 'ksc\_set' does not write values to the database. The scope of the variable that is set is valid from when the variable is defined to the end of the command steps for the entity. This make using 'ksc\_set' more attractive than using shell variables because the values are retained between separate 'ksc\_connect' sessions. Another advantage of using 'ksc\_set' is that the token values are visible in the logs, not just the variable names. This command may be nested within a 'ksc\_connect' command (see example below).

*Example using ksc\_set*

```
# Set the value of a compile flag.
#
ksc_set COMPILE="YES"
# ksc_set nested within a ksc_connect
ksc_connect dest_server
ksc_set REBUILD="NO"
ksc_exit
```

Later a temporary variable can be referenced in a command condition or in another command step. For example, the command condition may look like:

```
`[COMPILE]' = 'YES'
```

**ksc\_set\_env**

This command is used to set the correct environment context of an execution in cases where the Workflow source and destination environments are overridden using the DEST\_ENV and SOURCE\_ENV parameters. Normally it is not necessary to use this command because it is called internally from other Special Commands. If it is used on a stand alone basis, it must come after any 'ksc\_copy' commands.

Table A-14. ksc\_set\_env Parameters

| Parameter   | Default Token             | Description   |
|-------------|---------------------------|---|
| DEST_ENV_ID | [DEST_ENV.ENVIRONMENT_ID] | ID of the destination environment to be used instead of the destination environment on the current Workflow Step. |

Table A-14. *ksc\_set\_env* Parameters

| Parameter     | Default Token                   | Description   |
|---------------|---------------------------------|---|
| SOURCE_ENV_ID | [SOURCE_ENV.ENVIRONM<br>ENT_ID] | ID of the source environment to be used instead of the source environment on the current Workflow Step.             |
| SOURCE_ENV    | [SOURCE_ENV]                    | Name of the source environment to be used instead of the source environment on the current Workflow Step.           |
| DEST_ENV      | [DEST_ENV]                      | Name of the destination environment to be used instead of the destination environment on the current Workflow Step. |

## ksc\_store

This command dynamically sets the values of fields defined for Object Types, Request Types, and Report Types. This command is useful to set or alter the value of fields based on the command output. This command may only be used on fields which have been custom configured. Custom configured fields are those with tokens that are evaluated using the [P.<TOKEN>] or [VP.<TOKEN>] format. After altering a token, future evaluations of the token will use the new value. The new values are written to the database, so the changes are not temporary as in 'ksc\_set'.

This command may be nested within a 'ksc\_connect' command (as seen in the example below) and its value can be referenced in command conditions.

The following syntax is supported:

```
ksc_store TOKEN="Value"
ksc_store TOKEN="Hidden Value", "Visible Value"
```

In the first case, the hidden and visible values of the field will be set to the same value. In the second case, the hidden and visible values are set independently. "Hidden Value" refers to the [P.<TOKEN>] format. "Visible Value" refers to the [VP.<TOKEN>] format.

### Example using ksc\_store

In the example below, it is assumed that the entity in question has the following tokens defined:

```
[P.DRIVER]
[P.REVISION]
[P.RESULT]

# Store the name of the driver file.
```



```
ksc_store DRIVER="driver.sh"

# Capture the Revision number of a file.
#
ksc_connect_dest_server
cd "SourceCode/java"
grep '$Revision' ServerAdmin.java
ksc_store REVISION=" [EXEC.OUTPUT]"
ksc_exit

# Set the hidden and visible result codes of a parameter.
#
ksc_store RESULT="IN_PROG","In Progress"
```

## **ksc\_comment**

This command adds single line comments to the execution log. It can be used to indicate informational or error messages. HTML tags are supported.

The following syntax is supported:

```
ksc_comment <comment>
```

The comment text can be any text string.

## **ksc\_concsub**

This command submits Oracle Application concurrent requests from the operating system command line. It is treated as a special command because the command engine must capture the concurrent request id which is an output of successful submission. To work properly, this command must be called within a 'ksc\_connect - ksc\_exit' command block.

If 'ksc\_concsub' is used to submit a concurrent request to an Oracle Applications database other than the one the Kintana suite is currently installed on, the ORA\_APPS\_DB\_LINK parameter must be added to the 'ksc\_concsub' command. Otherwise, the status of the concurrent request cannot be determined after submission.

The following syntax is supported:

```
ksc_concsub ORA_APPS_DB_LINK="DB_LINK" CONCSub
```

where DB\_LINK corresponds to the database link from the Kintana schema to the APPS schema of the database to which the concurrent request is submitted.

### *Example using ksc\_concsub*

```
ksc_concsub ORA_APPS_DB_LINK=[DEST_ENV.ORA_APPS_DB_LINK]
CONCSUB
[DEST_ENV.APP.DB_USERNAME] / [DEST_ENV.APP.DB_PASSWORD]@[DE
ST_ENV.DB_CONNECT_STRING] FND 'Application Developer'
SYSADMIN WAIT=N CONCURRENT FND FNDFMREG
[DEST_ENV.APP_CODE] [P.P_FILENAME]
```



Note

The special command 'ksc\_concsub' is followed by the exact CONCSUB call that will be executed directly at the command line.

The complete syntax for Oracle's CONCSUB is shown below. Optional parameters are in square brackets.

```
CONCSUB
<ORACLE ID>
<Responsibility Application Short Name>
<Responsibility Name>
<User Name>
[WAIT=N]
CONCURRENT
<Concurrent Program Application Short Name>
<Concurrent Program Name>
[START=<Requested Start Date>]
[REPEAT_DAYS=<Repeat Interval>]
[REPEAT_END=<Request Resubmission End Date>]
<Concurrent Program Arguments...>
```

For additional information on using the CONCSUB command, see Oracle documentation.



Note

It is not possible to retrieve the concurrent request logs from a 'ksc\_concsub' submission submitted against a remote database.

## ksc\_begin\_script / ksc\_end\_script

The object command structure of the Kintana Product Suite lends itself nicely to standard, step-by-step processes. In most cases, these commands are fully capable of automating the migration of an object. However, in some circumstances it is necessary to add additional logic to the commands for an object. For example, perhaps a loop must be generated to repeat a command several times. This is where scripts-on-the-fly are best applied.

Scripts-on-the-fly are designed to leverage the architecture, tools, and knowledge already present in an organization. By using a script-on-the-fly, Kintana administrators can define migration logic in their preferred scripting language (such as Bourne Shell, C Shell or Perl). The scripts only need to be defined once. The Kintana execution engine copies the script wherever it needs

to be executed. The execution engine can also be instructed to clean up the script after it has been executed, leaving no traces behind.

The following syntax is supported:

```
ksc_begin_script <full_path_to_file_to_be_generated>
<directives from any scripting language>
ksc_end_script
```

It is commonly used in the following format:

```
ksc_begin_script [AS.PKG_TRANSFER_PATH] [P.P_SCRIPT_FILENAME]
```

because the script will be generated into a temporary directory by use of the [AS.PKG\_TRANSFER\_PATH] token. This token will reference a unique temporary directory per execution and ends with the proper directory slash '/' or '\'. After generation, the script can be transferred to another machine for execution using the 'ksc\_copy\_script' commands described later.

### *Example using ksc\_begin\_script and ksc\_end\_script*

```
ksc_begin_script [AS.PKG_TRANSFER_PATH] [P.P_SCRIPT_FILENAME]
#!/usr/bin/csh
#
# Script to lock, check in, and re-checkout the original
# file using RCS commands.
#
# Print a warning if the file does not exist.
#

if ($#argv != 2) then
    echo "$0 : wrong number of arguments"
    echo "Usage: $0 sub_path filename"
    exit 1
endif

set sub_path = $argv[1]
set filename = $argv[2]

if (-e "$sub_path/RCS/$filename,v") then
    rcs -l $sub_path/$filename
    ci -m"Before Copy." $sub_path/$filename
    co -l $sub_path/$filename
else
    echo "Warning: File $sub_path/$filename not found in
    RCS repository"
endif

exit 0
ksc_end_script

# Copy the script to the destination server and excute it.
ksc_copy_script_dest_server
ksc_connect_dest_server
csh [P.P_SCRIPT_FILENAME]
```

```
rm [P.P_SCRIPT_FILENAME]
ksc_exit
```

## ksc\_copy\_script Special Commands

These Special Commands are used to transfer files from the temporary file transfer directory (defined by token [AS.PKG\_TRANSFER\_PATH]) to other machines. These commands are typically used in conjunction with the ‘ksc\_begin\_script’ and ‘ksc\_end\_script’ commands, but can also be used in other ways.

### *ksc\_copy\_script\_dest\_client*

This command copies a script contained in [AS.PKG\_TRANSFER\_PATH], a temporary directory located on the Kintana server, to the base path of the destination client environment.

Table A-15. *ksc\_copy\_script\_dest\_client* Parameters

| Parameters      | Default Token               | Description  |
|-----------------|-----------------------------|--|
| SCRIPT_FILENAME | [P.P_SCRIPT_FILENAME]       | The name of the script file to transfer.   |
| DEST_BASE_PATH  | [DEST_ENV.CLIENT_BASE_PATH] | The base path of the destination client environment to be used instead of what is defined for the current destination environment. |
| DEST_ENV        | [DEST_ENV]                  | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

### *ksc\_copy\_script\_dest\_server*

This command copies a script contained in [AS.PKG\_TRANSFER\_PATH], a temporary directory located on the Kintana server, to the base path of the destination server environment.

Table A-16. *ksc\_copy\_script\_dest\_server* Parameters

| Parameters      | Default Token         | Description                              |
|-----------------|-----------------------|--|
| SCRIPT_FILENAME | [P.P_SCRIPT_FILENAME] | The name of the script file to transfer. |

Table A-16. *ksc\_copy\_script\_dest\_server* Parameters

| Parameters     | Default Token               | Description  |
|----------------|-----------------------------|--|
| DEST_BASE_PATH | [DEST_ENV.SERVER_BASE_PATH] | The base path of the destination server environment to be used instead of what is defined for the current destination environment. |
| DEST_ENV       | [DEST_ENV]                  | Name of the destination environment to be used instead of the destination environment on the current Workflow Step.                |

*ksc\_copy\_script\_source\_client*

This command copies a script contained in [AS.PKG\_TRANSFER\_PATH], a temporary directory located on the Kintana server, to the base path of the source client environment.

Table A-17. *ksc\_copy\_script\_source\_client* Parameters

| Parameters      | Default Token                 | Description  |
|-----------------|-------------------------------|--|
| SCRIPT_FILENAME | [P.P_SCRIPT_FILENAME]         | The name of the script file to transfer.   |
| DEST_BASE_PATH  | [SOURCE_ENV.CLIENT_BASE_PATH] | The base path of the source client environment to be used instead of what is defined for the current source environment. |
| SOURCE_ENV      | [SOURCE_ENV]                  | Name of the source environment to be used instead of the destination environment on the current Workflow Step.           |

*ksc\_copy\_script\_source\_server*

This command copies a script contained in [AS.PKG\_TRANSFER\_PATH], a temporary directory located on the Kintana server, to the base path of the source server environment.

Table A-18. *ksc\_copy\_script\_source\_client* Parameters

| Parameters      | Default Token         | Description                              |
|-----------------|-----------------------|--|
| SCRIPT_FILENAME | [P.P_SCRIPT_FILENAME] | The name of the script file to transfer. |

Table A-18. *ksc\_copy\_script\_source\_client* Parameters

| Parameters       | Default Token                     | Description  |
|------------------|-----------------------------------|--|
| SOURCE_BASE_PATH | [SOURCE_ENV.<br>SERVER_BASE_PATH] | The base path of the source server environment to be used instead of what is defined for the current source environment. |
| SOURCE_ENV       | [SOURCE_ENV]                      | Name of the source environment to be used instead of the source environment on the current Workflow Step.                |

## ksc\_om\_migrate

This command is used to launch migrations supported by the Object\*Migrator.

The following syntax is supported:

```
ksc_om_migrate CONC_PROGRAM=<conc_program_name>
APP_SHORT_NAME=<APP_SHORT_NAME> OM_ARCHIVE_FLAG=<Y/N>
```

Parameters CONC\_PROGRAM and APP\_SHORT\_NAME are required. All other parameters are optional and are used to override the default behavior.

Table A-19. *ksc\_om\_migrate* Parameters

| Parameter       | Default Token                        | Description  |
|-----------------|--------------------------------------|--|
| CONC_PROGRAM    | None. This is a mandatory parameter. | The concurrent program name. This has been pre-configured and will not need to be modified.  |
| OM_ARCHIVE_FLAG | [WFS.<br>OM_ARCHIVE_FLAG]            | Specifies whether the migration will store to the archive rather than using what has been specified for the current Workflow Step. |
| APP_SHORT_NAME  | None. This is a required parameter.  | This value is normally "CLM" but can be modified if the Object*Migrator has been installed into a custom account.                  |
| SOURCE_ENV      | [SOURCE_ENV]                         | The environment to migrate from rather than the one defined on the Workflow Step.  |
| DEST_ENV        | [DEST_ENV]                           | The environment to migrate to rather than the one defined on the Workflow Step.  |

## Example using `ksc_om_migrate`

```
#
#Launch an AOL Concurrent Program Migration
#
ksc_om_migrate CONC_PROGRAM="CLMRMCP1" APP_SHORT_NAME="CLM"
```

## `ksc_capture_output`

The ‘`ksc_capture_output`’ Special Command is only used in Validations. It is used to get data from an alternate source, and use that data to populate an auto-complete field. This functionality provides additional flexibility when designing auto-complete lists.

Many enterprises have found that they need to use alternate sources of data within their applications. Examples of these sources might be a flat file, an alternate database source, or output from a command line execution. The ‘`ksc_capture_output`’ command may be used in conjunction with these alternate data sources, in the context of a Validation, to provide a list of values on the fly.

The syntax for the ‘`ksc_capture_output`’ is:

```
ksc_capture_output <command>
```

In the Validation Workbench, under Validated By, choose either Command With Delimited Output or Command With Fixed Width Output and input the delimiting character or field length information. Then, under New Command, enter in the steps. The example below would put the validations into the `address.txt` file, then runs the ‘`ksc_capture_output`’ against the `address.txt` file:

```
ksc_begin_script [AS.PKG_TRANSFER_PATH] address.txt
street
city
state
zipcode
ksc_end_script
ksc_capture_output cat [AS.PKG_TRANSFER_PATH] address.txt
```

In the above scenario, the entire sequence of commands would be executed on the local machine where the Kintana server is running. This is the preferred method of invoking ‘`ksc_capture_output`’. The ‘`ksc_capture_output`’ command may be embedded between ‘`ksc_connect`’ and ‘`ksc_exit`’ commands, but the time delay is significant depending on network load (because the validation actually requires an entire TELNET, SSH or SSH2 session to be generated to the remote machine). It is recommended that ‘`ksc_capture_output`’ only be used in a local execution scenario.

'ksc\_capture\_output' may be called more than once. Each call will append the results to the previous call.

## ksc\_gl\_migrate

This command is used to launch migrations supported by the GL \*Migrator.

The following syntax is supported:

```
ksc_gl_migrate CONC_PROGRAM=<conc_program_name>
APP_SHORT_NAME=<APP_SHORT_NAME> GL_ARCHIVE_FLAG=<Y/N>
```

Parameters CONC\_PROGRAM and APP\_SHORT\_NAME are required. All other parameters are optional and are used to override the default behavior.

Table A-20. ksc\_gl\_migrate Parameters

| Parameter       | Default Token                        | Description  |
|-----------------|--------------------------------------|--|
| CONC_PROGRAM    | None. This is a mandatory parameter. | The concurrent program name. This has been pre-configured and will not need to be modified.                                      |
| GL_ARCHIVE_FLAG | [WFS.OM_ARCHIVE_FLAG]                | Specify whether the migration will store to the archive rather than using what has been specified for the current Workflow Step. |
| APP_SHORT_NAME  | None. This is a required parameter.  | This value is normally "CLGM" but can be modified if the GL *Migrator has been installed into a custom account.                  |
| SOURCE_ENV      | [SOURCE_ENV]                         | The environment to migrate from rather than the one defined on the Workflow Step.  |
| DEST_ENV        | [DEST_ENV]                           | The environment to migrate to rather than the one defined on the Workflow Step.  |

### Example ksc\_gl\_migrate

```
#
# Launch a Budget Organization migration
#
ksc_gl_migrate CONC_PROGRAM="CLGMRBO1" APP_SHORT_NAME="CLGM"
```



## ksc\_parse\_jcl

This command is only used by the 'OS/390 JCL Migration' Object Type to parse a JCL script using the Mainframe parameters for the specified environment.

Table A-21. *ksc\_parse\_jcl* Parameters

| Parameter | Default Token                 | Description  |
|-----------|-------------------------------|--|
| FILENAME  | [P.P_FILENAME]                | Name of the JCL source file to be edited.                                    |
| OUTFILE   | [OUTFILE]                     | Name of the output JCL file after applying the substitution expressions.     |
| ENV_ID    | [DEST_ENV.<br>ENVIRONMENT_ID] | The ID of the environment containing the mainframe substitution expressions. |

## ksc\_submit\_job

This command is only used by the 'OS/390 JCL Migration' Object Type to submit JCL to the Mainframe JES.

Table A-22. *ksc\_submit\_job* Parameters

| Parameter | Default Token              | Description                               |
|-----------|----------------------------|---|
| PATH      | [AS.PKG_TRANSFER_<br>PATH] | Path to the JCL file.                     |
| FILENAME  | [P.P_FILENAME]             | Name of the JCL source file to be edited. |

## ksc\_set\_exit\_value

This command is used to set the exit value of the command execution to any value. When not used, the command execution engine returns standard execution results, such as FAILURE, SUCCESS, and ERROR (if an internal error occurred) that the Workflow engine can transition on. Using 'ksc\_set\_exit\_value' gives the flexibility to set any exit value and allow custom Workflow transitions.

The following formats are supported:

```
# Sets the hidden and visible value to <value>.
ksc_set_exit_value "<value>"
```

```
# Sets both the hidden and visible values independently.
ksc_set_exit_value "<hidden_value>", "<visible_value>"
```

The Workflow engine will key off of the hidden value to determine if a transition should be made. The visible value is for display purposes.

'ksc\_set\_exit\_value' is ideal for situations where there could be a number of different execution results, not just Success or Failure. Using 'ksc\_set\_exit\_value' allows the Workflow engine to transition on any number of execution outcomes.

## ksc\_clear\_exit\_value

This command is used to clear the exit value set by 'ksc\_set\_exit\_value'. When cleared, the execution engine will return its standard results, SUCCESS, FAILURE, or ERROR.

## ksc\_run\_sql

This command runs a SQL query against the chosen Environment. The result of the last row queried is returned in the [SQL\_OUTPUT] token. The result of the entire query is put into the file [AS.PKG\_TRANSFER\_PATH][PKGL.SEQ].txt in the \$KNTA\_HOME.

To run this Special Command, any Execution Steps in the Deliver Workflow must have their Source Environments defined. You can do this in the Workflow Step window.

The Special Command's parameters are described in [Table A-23](#).

Table A-23. ksc\_run\_sql Parameters

| Parameter        | Default Token      | Description  |
|------------------|--------------------|--|
| QUERY_STRING     | [QUERY_STRING]     | A SQL select statement.  |
| ENV_NAME         | [ENV_NAME]         | The name of the environment from where you want to query data. The JDBC connection should be checked in the environment checker. |
| EXCEPTION_OPTION | [EXCEPTION_OPTION] | If no data is returned, determines if an exception should be thrown. The only available option is '-no_data_exception.'          |

Example `ksc_run_sql`

```
ksc_run_sql QUERY_STRING="select sysdate from sys.dual"
ENV_NAME=" [SOURCE_ENV.ENVIRONMENT_NAME] "
```



Note

The 'ksc\_run\_sql' special command can be used to populate a Validation. This is appropriate when the Validation is validated by a Command with Delimited Output. In this case, the Data Delimiter should be set to "#@#".

The following code is an example of `ksc_run_sql` in a Validation.

```
ksc_run_sql QUERY_STRING="select id, name from some_table"
ENV_NAME=" [SOURCE_ENV.ENVIRONMENT_NAME] "
ksc_capture_output cat [AS.PKG_TRANSFER_PATH] [PKGL.SEQ].txt
```

# Summary of All Special Command Parameters

The following table provides the parameters for all predefined Kintana Special Commands.

Table A-24. Special Command Parameters

| Special Command         | Parameters          | Defaults                               |
|-------------------------|---------------------|--|
| ksc_begin_script        |                     |  |
| ksc_comment             |                     |  |
| ksc_concsub             |                     |  |
| ksc_connect_dest_client | USERNAME            | [DEST_ENV.CLIENT_USERNAME]             |
|                         | PASSWORD            | [DEST_ENV.CLIENT_PASSWORD]             |
|                         | NT_DOMAIN           | [DEST_ENV.CLIENT_NT_DOMAIN]            |
|                         | DEST_BASE_PATH      | [DEST_ENV.CLIENT_BASE_PATH]            |
|                         | CONNECTION_PROTOCOL | [DEST_ENV.CLIENT_CON_PROTOCOL_MEANING] |
|                         | DEST_ENV            | [DEST_ENV]                             |

Table A-24. Special Command Parameters

| Special Command           | Parameters          | Defaults                                 |
|---------------------------|---------------------|--|
| ksc_connect_dest_server   | USERNAME            | [DEST_ENV.SERVER_USERNAME]               |
|                           | PASSWORD            | [DEST_ENV.SERVER_PASSWORD]               |
|                           | NT_DOMAIN           | [DEST_ENV.SERVER_NT_DOMAIN]              |
|                           | DEST_BASE_PATH      | [DEST_ENV.SERVER_BASE_PATH]              |
|                           | CONNECTION_PROTOCOL | [DEST_ENV.SERVER_CON_PROTOCOL_MEANING]   |
|                           | DEST_ENV            | [DEST_ENV]                               |
| ksc_connect_source_client | USERNAME            | [SOURCE_ENV.CLIENT_USERNAME]             |
|                           | PASSWORD            | [SOURCE_ENV.CLIENT_PASSWORD]             |
|                           | NT_DOMAIN           | [SOURCE_ENV.CLIENT_NT_DOMAIN]            |
|                           | SOURCE_BASE_PATH    | [SOURCE_ENV.CLIENT_BASE_PATH]            |
|                           | CONNECTION_PROTOCOL | [SOURCE_ENV.CLIENT_CON_PROTOCOL_MEANING] |
|                           | SOURCE_ENV          | [SOURCE_ENV]                             |
| ksc_connect_source_server | USERNAME            | [SOURCE_ENV.SERVER_USERNAME]             |
|                           | PASSWORD            | [SOURCE_ENV.SERVER_PASSWORD]             |
|                           | NT_DOMAIN           | [SOURCE_ENV.SERVER_NT_DOMAIN]            |
|                           | SOURCE_BASE_PATH    | [SOURCE_ENV.SERVER_BASE_PATH]            |
|                           | CONNECTION_PROTOCOL | [SOURCE_ENV.SERVER_CON_PROTOCOL_MEANING] |
|                           | SOURCE_ENV          | [SOURCE_ENV]                             |
| ksc_copy_client_client    | SUB_PATH            | [P.P.SUB_PATH]                           |
|                           | SOURCE_BASE_PATH    | [SOURCE_ENV.CLIENT_BASE_PATH]            |
|                           | DEST_BASE_PATH      | [DEST_ENV.CLIENT_BASE_PATH]              |
|                           | FILENAME            | [P.P.FILENAME]                           |
|                           | FILE_TYPE           | [P.P.FILE_TYPE]                          |
|                           | SOURCE_ENV          | [SOURCE_ENV]                             |
|                           | DEST_ENV            | [DEST_ENV]                               |

Table A-24. Special Command Parameters

| Special Command        | Parameters       | Defaults                      |
|------------------------|------------------|-------------------------------|
| ksc_copy_client_server | SUB_PATH         | [P.P.SUB_PATH]                |
|                        | SOURCE_BASE_PATH | [SOURCE_ENV.CLIENT_BASE_PATH] |
|                        | DEST_BASE_PATH   | [DEST_ENV.SERVER_BASE_PATH]   |
|                        | FILENAME         | [P.P.FILENAME]                |
|                        | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                        | SOURCE_ENV       | [SOURCE_ENV]                  |
|                        | DEST_ENV         | [DEST_ENV]                    |
| ksc_copy_server_client | SUB_PATH         | [P.P.SUB_PATH]                |
|                        | SOURCE_BASE_PATH | [SOURCE_ENV.SERVER_BASE_PATH] |
|                        | DEST_BASE_PATH   | [DEST_ENV.CLIENT_BASE_PATH]   |
|                        | FILENAME         | [P.P.FILENAME]                |
|                        | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                        | SOURCE_ENV       | [SOURCE_ENV]                  |
|                        | DEST_ENV         | [DEST_ENV]                    |
| ksc_copy_server_server | SUB_PATH         | [P.P.SUB_PATH]                |
|                        | SOURCE_BASE_PATH | [SOURCE_ENV.SERVER_BASE_PATH] |
|                        | DEST_BASE_PATH   | [DEST_ENV.SERVER_BASE_PATH]   |
|                        | FILENAME         | [P.P.FILENAME]                |
|                        | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                        | SOURCE_ENV       | [SOURCE_ENV]                  |
|                        | DEST_ENV         | [DEST_ENV]                    |

Table A-24. Special Command Parameters

| Special Command             | Parameters       | Defaults                      |
|-----------------------------|------------------|-------------------------------|
| ksc_copy_client_tmp         | SUB_PATH         | [P.P.SUB_PATH]                |
|                             | SOURCE_BASE_PATH | [SOURCE_ENV.CLIENT_BASE_PATH] |
|                             | DEST_BASE_PATH   | [DEST_ENV.CLIENT_BASE_PATH]   |
|                             | FILENAME         | [P.P.FILENAME]                |
|                             | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                             | SOURCE_ENV       | [SOURCE_ENV]                  |
| ksc_copy_server_tmp         | SUB_PATH         | [P.P.SUB_PATH]                |
|                             | SOURCE_BASE_PATH | [SOURCE_ENV.SERVER_BASE_PATH] |
|                             | FILENAME         | [P.P.FILENAME]                |
|                             | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                             | SOURCE_ENV       | [SOURCE_ENV]                  |
| ksc_copy_tmp_client         | SUB_PATH         | [P.P.SUB_PATH]                |
|                             | DEST_BASE_PATH   | [DEST_ENV.CLIENT_BASE_PATH]   |
|                             | FILENAME         | [P.P.FILENAME]                |
|                             | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                             | DEST_ENV         | [DEST_ENV]                    |
| ksc_copy_tmp_server         | SUB_PATH         | [P.P.SUB_PATH]                |
|                             | DEST_BASE_PATH   | [DEST_ENV.SERVER_BASE_PATH]   |
|                             | FILENAME         | [P.P.FILENAME]                |
|                             | FILE_TYPE        | [P.P.FILE_TYPE]               |
|                             | DEST_ENV         | [DEST_ENV]                    |
| ksc_copy_script_dest_client | SCRIPT_FILENAME  | [P.P.SCRIPT_FILENAME]         |
|                             | DEST_BASE_PATH   | [DEST_ENV.CLIENT.BASE_PATH]   |
|                             | DEST_ENV         | [DEST_ENV]                    |

Table A-24. Special Command Parameters

| Special Command               | Parameters       | Defaults                      |
|-------------------------------|------------------|-------------------------------|
| ksc_copy_script_dest_server   | SCRIPT_FILENAME  | [P.P_SCRIPT_FILENAME]         |
|                               | DEST_BASE_PATH   | [DEST_ENV.SERVER_BASE_PATH]   |
|                               | DEST_ENV         | [DEST_ENV]                    |
| ksc_copy_script_source_client | SCRIPT_FILENAME  | [P.P_SCRIPT_FILENAME]         |
|                               | SOURCE_BASE_PATH | [SOURCE_ENV.CLIENT_BASE_PATH] |
|                               | SOURCE_ENV       | [SOURCE_ENV]                  |
| ksc_copy_script_source_server | SCRIPT_FILENAME  | [P.P_SCRIPT_FILENAME]         |
|                               | SOURCE_BASE_PATH | [SOURCE_ENV.SERVER_BASE_PATH] |
|                               | SOURCE_ENV       | [SOURCE_ENV]                  |
| ksc_clear_exit_value          |                  |                               |
| ksc_end_script                |                  |                               |
| ksc_exit                      |                  |                               |
| ksc_gl_migrate                | CONC_PROGRAM     |                               |
|                               | SOURCE_ENV       | [SOURCE_ENV]                  |
|                               | DEST_ENV         | [DEST_ENV]                    |
|                               | GL_ARCHIVE_FLAG  | [WFS.GL_ARCHIVE_FLAG]         |
|                               | APP_SHORT_NAME   |                               |
| ksc_local_exec                |                  |                               |
| ksc_om_migrate                | CONC_PROGRAM     |                               |
|                               | SOURCE_ENV       | [SOURCE_ENV]                  |
|                               | DEST_ENV         | [DEST_ENV]                    |
|                               | OM_ARCHIVE_FLAG  | [WFS.OM_ARCHIVE_FLAG]         |
|                               | APP_SHORT_NAME   |                               |
| ksc_parse_jcl                 | FILENAME         | [P.P_FILENAME]                |
|                               | OUTFILE          |                               |
|                               | ENV_ID           | [DEST_ENV.ENVIRONMENT_ID]     |

Table A-24. Special Command Parameters

| Special Command    | Parameters    | Defaults                    |
|--------------------|---------------|-----------------------------|
| ksc_replace        | FILENAME      | [P.P_FILENAME]              |
|                    | OUTFILE       |                             |
|                    | SUBST         |                             |
| ksc_respond        |               |                             |
| ksc_run_sql        |               |                             |
| ksc_set            | Custom Token  |                             |
| ksc_set_env        | DEST_ENV_ID   | [DEST_ENV.ENVIRONMENT_ID]   |
|                    | SOURCE_ENV_ID | [SOURCE_ENV.ENVIRONMENT_ID] |
|                    | SOURCE_ENV    | [SOURCE_ENV]                |
|                    | DEST_ENV      | [DEST_ENV]                  |
| ksc_set_exit_value |               |                             |
| ksc_simple_respond |               |                             |
| ksc_store          | Custom Token  |                             |
| ksc_submit_job     | PATH          | [AS.PKG_TRANSFER_PATH]      |
|                    | FILENAME      | [P.P_FILENAME]              |



# Appendix B Tokens

This appendix provides a list of all entity Tokens in Kintana. Use the following table as a quick reference guide to jump to the desired location.

Table B-1. Token Tables

| Table                                      | Page       |
|--|------------|
| <i>Table B-2, App Server Properties</i>    | <i>96</i>  |
| <i>Table B-3, Budget</i>                   | <i>97</i>  |
| <i>Table B-4, Contacts</i>                 | <i>97</i>  |
| <i>Table B-5, Distribution</i>             | <i>98</i>  |
| <i>Table B-6, Environments</i>             | <i>99</i>  |
| <i>Table B-7, Environment Applications</i> | <i>101</i> |
| <i>Table B-8, Command Execution</i>        | <i>102</i> |
| <i>Table B-9, Notifications</i>            | <i>103</i> |
| <i>Table B-10, Organization Unit</i>       | <i>104</i> |
| <i>Table B-11, Packages</i>                | <i>104</i> |
| <i>Table B-12, Package Lines</i>           | <i>106</i> |
| <i>Table B-13, Package Pending</i>         | <i>107</i> |
| <i>Table B-14, Program</i>                 | <i>108</i> |
| <i>Table B-15, Projects</i>                | <i>108</i> |
| <i>Table B-16, Project Details</i>         | <i>110</i> |
| <i>Table B-17, Releases</i>                | <i>110</i> |
| <i>Table B-18, Requests</i>                | <i>111</i> |

Table B-1. Token Tables

| Table  | Page       |
|--|------------|
| <i>Table B-1, Request Details</i>            | <i>113</i> |
| <i>Table B-19, Request Pending</i>           | <i>113</i> |
| <i>Table B-20, Report Submissions</i>        | <i>114</i> |
| <i>Table B-21, Resource Pools</i>            | <i>115</i> |
| <i>Table B-22, Security Groups</i>           | <i>116</i> |
| <i>Table B-23, Skill</i>                     | <i>116</i> |
| <i>Table B-24, Staffing Profile</i>          | <i>117</i> |
| <i>Table B-25, System</i>                    | <i>117</i> |
| <i>Table B-26, Tasks</i>                     | <i>118</i> |
| <i>Table B-27, Tasks Pending</i>             | <i>120</i> |
| <i>Table B-28, Users</i>                     | <i>121</i> |
| <i>Table B-29, Validations</i>               | <i>122</i> |
| <i>Table B-30, Validation Values</i>         | <i>123</i> |
| <i>Table B-31, Workflows</i>                 | <i>123</i> |
| <i>Table B-32, Workflow Steps</i>            | <i>124</i> |
| <i>Table B-33, Workflow Step Transaction</i> | <i>126</i> |

See “*Field Group Tokens*” on page 126 for a list of the tokens that are associated with Kintana’s Field Groups.

## Kintana System Tokens

Table B-2. App Server Properties

| Prefix | Token             | Description   |
|--------|-------------------|---|
| AS     | PKG_TRANSFER_PATH | Temporary directory used for files during command executions. |



Note

Other App Server Properties tokens are generated from the parameters in the server.conf file. See the Server Parameter appendix in the “*Kintana System Administration Guide*” for a description of each server parameter.

Table B-3. Budget

| Prefix | Token                     | Description  |
|--------|---------------------------|--|
| BGT    | ACTIVE_FLAG               | The active flag of the Budget.   |
| BGT    | BUDGET_ID                 | The ID of the Budget (defined in the table KCST_BUDGETS).                      |
| BGT    | BUDGET_IS_FOR_ENTITY_NAME | The entity name (Project, Program, or Org Unit) to which the Budget is linked. |
| BGT    | BUDGET_IS_FOR_ID          | The ID of the Project/Program/Org Unit to which the Budget is linked.          |
| BGT    | BUDGET_IS_FOR_NAME        | The name of the Project/Program/Org Unit to which the Budget is linked.        |
| BGT    | BUDGET_NAME               | The name of the Budget.  |
| BGT    | BUDGET_ROLLS_UP_TO_ID     | The ID of the Budget to which this Budget rolls up to.                         |
| BGT    | BUDGET_ROLLS_UP_TO_NAME   | The name of the Budget to which this Budget rolls up to.                       |
| BGT    | BUDGET_URL                | The URL to view this Budget.   |
| BGT    | CREATED_BY                | The username of the user who created the Budget.                               |
| BGT    | CREATION_DATE             | The date when the Budget was created.  |
| BGT    | DESCRIPTION               | The description of the Budget.   |
| BGT    | END_PERIOD                | The end period of the Budget.  |
| BGT    | INITIATION_REQ            | The initiation Request ID of the Budget.                                       |
| BGT    | PERIOD_SIZE               | The period size of the Budget.   |
| BGT    | START_PERIOD              | The start period of the Budget.  |
| BGT    | STATUS_CODE               | The status code of the Budget.   |
| BGT    | STATUS_NAME               | The status name of the Budget.   |

Table B-4. Contacts

| Prefix | Token         | Description   |
|--------|---------------|---|
| CON    | COMPANY       | The company the Contact works for.                          |
| CON    | COMPANY_NAME  | The name of the company the Contact works for.              |
| CON    | CONTACT_ID    | The ID of the Contact (defined in the table KCRT_CONTACTS). |
| CON    | CREATED_BY    | The ID of the User that created the Contact.                |
| CON    | CREATION_DATE | The date the Contact was created.                           |
| CON    | EMAIL_ADDRESS | The email address of the Contact.                           |
| CON    | FIRST_NAME    | The first name of the Contact.                              |

Table B-4. Contacts

| Prefix | Token            | Description  |
|--------|------------------|--|
| CON    | FULL_NAME        | The full name of the Contact.  |
| CON    | LAST_NAME        | The last name of the Contact.  |
| CON    | LAST_UPDATED_BY  | The ID of the User that last updated the Contact.  |
| CON    | LAST_UPDATE_DATE | The date the Contact was last updated.   |
| CON    | PHONE_NUMBER     | The phone number of the Contact.   |
| CON    | USER_ID          | The UserID of the Contact, if the Contact is a Kintana user.   |
| CON    | USERNAME         | The username of the Contact (if applicable). This may be a username for an external system, not necessarily Kintana. |

Table B-5. Distribution

| Prefix | Tokens                   | Description   |
|--------|--------------------------|---|
| DIST   | CREATED_BY               | The ID of the user that created the Distribution.   |
| DIST   | CREATED_BY_USERNAME      | The Kintana Username of the user that created the Distribution.                                 |
| DIST   | DESCRIPTION              | The description of the Release.   |
| DIST   | DISTRIBUTION_ID          | The ID of the Distribution (defined in table KREL_DISTRIBUTION).                                |
| DIST   | DISTRIBUTION_NAME        | The name of the Distribution.   |
| DIST   | DISTRIBUTION_STATUS      | The Workflow status of the Distribution Workflow.   |
| DIST   | FEEDBACK_FLAG            | Whether the Distribution has fed back a specified value to the Package Lines being distributed. |
| DIST   | FEEDBACK_VALUE           | The value to be returned to the original Package Lines.   |
| DIST   | LAST_UPDATED_BY          | The ID of the user that last updated the Distribution.  |
| DIST   | LAST_UPDATED_BY_USERNAME | The Kintana username of the user that last updated the Distribution.                            |
| DIST   | LAST_UPDATE_DATE         | The date the Distribution was last updated.   |
| DIST   | RELEASE_ID               | The ID of the Release that created this Distribution.   |
| DIST   | RELEASE_NAME             | The name of the Release that created this Distribution.   |
| DIST   | WORKFLOW                 | The Workflow used to process the Distribution.  |

Table B-6. Environments

| Prefix | Token                            | Description   |
|--------|----------------------------------|---|
| ENV    | CLIENT_BASE_PATH                 | The base (root) path of the client.   |
| ENV    | CLIENT_CON_PROTOCOL              | The protocol used to connect to this client.  |
| ENV    | CLIENT_CON_PROTOCOL_MEANING      | The visible value of the client connect protocol.   |
| ENV    | CLIENT_NAME                      | The DNS name or IP address of the client computer.  |
| ENV    | CLIENT_NT_DOMAIN                 | The domain name for the client, if the client machine type is Windows.                                    |
| ENV    | CLIENT_ENABLED_FLAG              | The flag indicating whether the client portion of the Environment is enabled.                             |
| ENV    | CLIENT_PASSWORD                  | The password Kintana uses to log onto or access the client. This value is encrypted.                      |
| ENV    | CLIENT_TYPE_CODE                 | The Validation value code of the client machine type.   |
| ENV    | CLIENT_USERNAME                  | The username Kintana uses to log onto or access the client.   |
| ENV    | CLIENT_TRANSFER_PROTOCOL         | The protocol used to transfer files to or from this client.   |
| ENV    | CLIENT_TRANSFER_PROTOCOL_MEANING | The visible value of the client transfer protocol.  |
| ENV    | CREATED_BY                       | The ID of the User that created the Environment.  |
| ENV    | CREATION_DATE                    | The date the environment was created.   |
| ENV    | DATABASE_ENABLED_FLAG            | The flag indicating whether the database portion of the Environment is enabled.                           |
| ENV    | DATABASE_TYPE                    | The Validation value code of the database type.   |
| ENV    | DB_CONNECT_STRING                | For Oracle database type, the connect string used to access the database from the command line.           |
| ENV    | DB_LINK                          | For Oracle database type, the database link from the Kintana schema to the Environment's database schema. |
| ENV    | DB_NAME                          | The DNS name or IP address of the database server.  |
| ENV    | DB_ORACLE_SID                    | For Oracle database type, the SID of the database (often the same as the DB_CONNECT_STRING).              |

Table B-6. Environments [continued]

|     |                                  |  |
|-----|----------------------------------|--|
| ENV | DB_PASSWORD                      | The password Kintana uses to log onto or access the database. This value is encrypted.                                     |
| ENV | DB_PORT_NUMBER                   | For Oracle database type, the port number on which SQL*Net is listening for remote SQL connections on the database server. |
| ENV | DB_USERNAME                      | The username or schema name Kintana uses to log onto or access the database.   |
| ENV | DB_VERSION                       | The version of the database (such as 8.1.7).   |
| ENV | DESCRIPTION                      | The description of the Environment.  |
| ENV | ENABLED_FLAG                     | The flag indicating whether the Environment is enabled and available for use in Workflows.                                 |
| ENV | ENVIRONMENT_ID                   | The ID of the Environment in the table KENV_ENVIRONMENTS.  |
| ENV | ENVIRONMENT_NAME                 | The name of the Environment.   |
| ENV | LAST_UPDATED_BY                  | The ID of the User that last updated the Environment.  |
| ENV | LAST_UPDATE_DATE                 | The date the Environment was last updated.   |
| ENV | LOCATION                         | The location of the Environment.   |
| ENV | MSSQL_DB_NAME                    | For MS SQL Server database type, the database name used to access the database from the command line.                      |
| ENV | SERVER_BASE_PATH                 | The base (root) path of the server.  |
| ENV | SERVER_CON_PROTOCOL              | The protocol used to connect to this server.   |
| ENV | SERVER_CON_PROTOCOL_MEANING      | The visible value of the server connection protocol.   |
| ENV | SERVER_TRANSFER_PROTOCOL         | The protocol used to transfer files to or from this server.  |
| ENV | SERVER_TRANSFER_PROTOCOL_MEANING | The visible value of the server transfer protocol.   |
| ENV | SERVER_ENABLED_FLAG              | The flag indicating whether the server portion of the Environment is enabled.  |
| ENV | SERVER_NAME                      | The DNS name or IP address of the server computer.   |
| ENV | SERVER_NT_DOMAIN                 | The domain name for the server, if the server machine type is Windows.   |

Table B-6. Environments [continued]

|     |                  |  |
|-----|------------------|--|
| ENV | SERVER_PASSWORD  | The password Kintana uses to log onto or access the server. This value is encrypted. |
| ENV | SERVER_TYPE_CODE | The Validation value code of the server machine type.                                |
| ENV | SERVER_USERNAME  | The username Kintana uses to log onto or access the server.                          |



Note

If you have installed any Kintana Accelerators, there will be more Environment tokens with the prefix 'AC.' Refer to the Accelerator documents available at <http://www.kintana.com/support/download/login.jsp> for more detailed information on these tokens.

Table B-7. Environment Applications

| Prefix  | Token                            | Description  |
|---------|----------------------------------|--|
| ENV.APP | APP_CODE                         | The short name (code) for the Application.   |
| ENV.APP | APP_NAME                         | The descriptive name for the Application.  |
| ENV.APP | CLIENT_BASE_PATH                 | The Application-specific base (root) path of the client.   |
| ENV.APP | CLIENT_PASSWORD                  | The Application-specific password Kintana uses to log onto or access the client. This value is encrypted.                      |
| ENV.APP | CLIENT_USERNAME                  | The Application-specific username Kintana uses to log onto or access the client.   |
| ENV.APP | CLIENT_CON_PROTOCOL              | The Application-specific protocol used to connect to this client.  |
| ENV.APP | CLIENT_CON_PROTOCOL_MEANING      | The visible value of the client connection protocol.   |
| ENV.APP | CLIENT_TRANSFER_PROTOCOL         | The application-specific protocol used to transfer files to and from this client.  |
| ENV.APP | CLIENT_TRANSFER_PROTOCOL_MEANING | The visible value of the client transfer protocol.   |
| ENV.APP | CREATED_BY                       | The ID of the User that created the Application.   |
| ENV.APP | CREATION_DATE                    | The date the Application was created.  |
| ENV.APP | DB_LINK                          | For Oracle database type, the Application-specific database link from the Kintana schema to the Environment's database schema. |
| ENV.APP | DB_NAME                          | For MS SQL Server database type, the Application-specific database name used to access the database from the command line.     |

Table B-7. Environment Applications

|         |                                  |   |
|---------|----------------------------------|---|
| ENV.APP | DB_PASSWORD                      | The Application-specific password Kintana uses to log onto or access the database. This value is encrypted. |
| ENV.APP | DB_USERNAME                      | The Application-specific username or schema name Kintana uses to log onto or access the database.           |
| ENV.APP | DESCRIPTION                      | The description of the Application.   |
| ENV.APP | ENABLED_FLAG                     | The flag indicating whether the Application is enabled and available for selection in Package Lines.        |
| ENV.APP | ENVIRONMENT_APP_ID               | The ID of the Application in the table KENV_ENVIRONMENT_APPS.   |
| ENV.APP | ENVIRONMENT_ID                   | The ID of the Environment the Application is associated with.   |
| ENV.APP | ENVIRONMENT_NAME                 | The name of the Environment the application is associated with.   |
| ENV.APP | LAST_UPDATED_BY                  | The ID of the User that last updated the Application.   |
| ENV.APP | LAST_UPDATE_DATE                 | The date the Application was last updated.  |
| ENV.APP | SERVER_CON_PROTOCOL              | The Application-specific protocol used to connect to this server.   |
| ENV.APP | SERVER_CON_PROTOCOL_MEANING      | The visible value of the server connection protocol.  |
| ENV.APP | SERVER_TRANSFER_PROTOCOL         | The application-specific protocol used to transfer files to and from this server.                           |
| ENV.APP | SERVER_TRANSFER_PROTOCOL_MEANING | The visible value of the server transfer protocol.  |
| ENV.APP | SERVER_BASE_PATH                 | The Application-specific base (root) path of the server   |
| ENV.APP | SERVER_PASSWORD                  | The Application-specific password Kintana uses to log onto or access the server. This value is encrypted.   |
| ENV.APP | SERVER_USERNAME                  | The Application-specific username Kintana uses to log onto or access the server.                            |
| ENV.APP | WORKBENCH_ENVIRONMENT_URL        | The URL of the Environment window in the Kintana Workbench.   |

Table B-8. Command Execution

| Prefix | Token     | Description                                       |
|--------|-----------|---|
| EXEC   | EXIT_CODE | The exit code of a Command execution.             |
| EXEC   | OUTPUT    | The last line of output from a Command execution. |





Note

The Command Execution tokens, [EXEC.OUTPUT] and [EXEC.EXIT\_CODE], can be used in the following contexts:

- Inside Command step segments that use the ksc\_connect and ksc\_exit Special Commands.
- Immediately after Command step segments that use the ksc\_local\_exec Special Command.

For example, the following code segment shows you how to use both Command Execution tokens to retrieve the output and exit code immediately upon execution. The tokens are used immediately after the ksc\_local\_exec Special Command.

```
ksc_local_exec pwd
ksc_set MY_PATH=" [EXEC.OUTPUT] "
ksc_set MY_EXIT_CODE=" [EXEC.EXIT_CODE] "
ksc_local_exec echo `[MY_PATH]/bin`
ksc_local_exec echo `[MY_EXIT_CODE]`
```

Table B-9. Notifications

| Prefix | Tokens               | Description  |
|--------|----------------------|--|
| NOTIF  | CC_USERS             | The list of users on the Cc: header of the Notification.                                       |
| NOTIF  | CHANGED_FIELD        | The field that changed to trigger a notification.  |
| NOTIF  | EXCEPTION_RULE       | The exception rule that was met by the task exception that caused the notification to be sent. |
| NOTIF  | EXCEPTION_RULE_NAME  | The name of the task exception that caused the notification to be sent.                        |
| NOTIF  | EXCEPTION_VIOLATION  | The specific violation of the exception that caused the notification to be sent.               |
| NOTIF  | NEW_VALUE            | The new value of the changed field.  |
| NOTIF  | NOTIFICATION_DETAILS | Notification details for linked tokens.  |
| NOTIF  | OLD_VALUE            | The previous value of the changed field.   |
| NOTIF  | TO_USERS             | The list of users on the To: header of the notification.                                       |

Table B-10. Organization Unit

| Prefix | Tokens               | Description  |
|--------|----------------------|--|
| ORG    | BUDGET_ID            | The ID of the Budget linked to this Org unit.                                    |
| ORG    | BUDGET_NAME          | The name of the Budget linked to this Org unit.                                  |
| ORG    | CATEGORY_CODE        | The lookup code of the Org unit category (lookup type = RSC - Org Unit Category) |
| ORG    | CATEGORY_NAME        | The category name of the Org unit.   |
| ORG    | CREATED_BY           | The ID of the user that created the Org unit.                                    |
| ORG    | CREATED_BY_USERNAME  | The name of the user that created the Org unit.                                  |
| ORG    | CREATION_DATE        | The date that the Org unit was created.  |
| ORG    | DEPARTMENT_CODE      | The lookup code of the Org unit department (lookup type = DEPT)                  |
| ORG    | DEPARTMENT_NAME      | The department name of the Org Unit.   |
| ORG    | LOCATION_CODE        | The lookup code of the Org Unit Location (lookup type = RSC - Location)          |
| ORG    | LOCATION_NAME        | The location name of the Org unit.   |
| ORG    | MANAGER_ID           | The ID of the manager of the Org unit.   |
| ORG    | MANAGER_USERNAME     | The name of the manager of the Org unit.   |
| ORG    | ORG_UNIT_ID          | The ID of the Org unit (defined in table KRSC_ORG_UNITS).                        |
| ORG    | ORG_UNIT_NAME        | The name of the Org unit.  |
| ORG    | PARENT_ORG_UNIT_ID   | The ID of the parent Org unit.   |
| ORG    | PARENT_ORG_UNIT_NAME | The name of the parent Org unit.   |

Table B-11. Packages

| Prefix | Token                  | Description   |
|--------|------------------------|---|
| PKG    | ASSIGNED_TO_EMAIL      | The email address of the User that the Package is assigned to.      |
| PKG    | ASSIGNED_TO_GROUP_ID   | The ID of the Security Group that the Package has been assigned to. |
| PKG    | ASSIGNED_TO_GROUP_NAME | The Security Group that the Package has been assigned to.           |

Table B-11. Packages

| Prefix | Token                    | Description   |
|--------|--------------------------|---|
| PKG    | ASSIGNED_TO_USERNAME     | The name of the User that the Package has been assigned to.                   |
| PKG    | ASSIGNED_TO_USER_ID      | The ID of the user that the Package has been assigned to.                     |
| PKG    | CREATED_BY               | The ID of the user that created the Package.                                  |
| PKG    | CREATED_BY_EMAIL         | The email address of the User that created the Package.                       |
| PKG    | CREATED_BY_USERNAME      | The Kintana username of the User that created the Package.                    |
| PKG    | CREATION_DATE            | The date the Package was created.   |
| PKG    | DESCRIPTION              | The description of the Package.   |
| PKG    | ID                       | The ID of the Package in the table KDLV_PACKAGES.                             |
| PKG    | LAST_UPDATED_BY          | The ID of the User that last updated the Package.                             |
| PKG    | LAST_UPDATED_BY_EMAIL    | The email address of the User that last updated the Package.                  |
| PKG    | LAST_UPDATED_BY_USERNAME | The Kintana username of the User that last updated the Package.               |
| PKG    | LAST_UPDATE_DATE         | The date the Package was last updated.  |
| PKG    | NOTES                    | The notes for the Package.  |
| PKG    | NUMBER                   | The name/number of the Package.   |
| PKG    | PACKAGE_GROUP_CODE       | The Package Group code.   |
| PKG    | PACKAGE_GROUP_NAME       | The name of the Package Group.  |
| PKG    | PARENT_REQUEST_ID        | The ID of the Request that created this Package (if applicable).              |
| PKG    | PRIORITY                 | The priority of the Package.  |
| PKG    | PRIORITY_CODE            | The Validation value code of the Package priority.                            |
| PKG    | PRIORITY_NAME            | The Validation value meaning of the Package priority.                         |
| PKG    | PRIORITY_SEQ             | The priority sequence of the Package.   |
| PKG    | PROJECT_CODE             | The Validation value code of the Project the Package belongs to.              |
| PKG    | PROJECT_NAME             | The Validation value meaning of the Project the Package belongs to.           |
| PKG    | SUBMIT_DATE              | The date that the Package was submitted.                                      |
| PKG    | REQUESTED_BY_EMAIL       | The email address of the User who requested the Package.                      |
| PKG    | REQUESTED_BY_USERNAME    | The Kintana username of the User who requested the Package.                   |
| PKG    | REQUESTED_BY_USER_ID     | The ID of the user that requested the Package.                                |
| PKG    | PACKAGE_ID               | The ID of the Package in the table KDLV_PACKAGES.                             |
| PKG    | PACKAGE_NO_LINK          | Shows up as a standard hyperlink to the Package in HTML-format Notifications. |

Table B-11. Packages

| Prefix | Token                     | Description   |
|--------|---------------------------|---|
| PKG    | PACKAGE_TYPE              | The Validation value meaning of the Package type.           |
| PKG    | PACKAGE_TYPE_CODE         | The Validation value code of the Package type.              |
| PKG    | PACKAGE_URL               | The URL of the Package in the Kintana interface.            |
| PKG    | PERCENT_COMPLETE          | Percent complete of the Package.                            |
| PKG    | RUN_GROUP                 | The run group of the Package.                               |
| PKG    | STATUS                    | The Validation value meaning for the status of the Package. |
| PKG    | STATUS_CODE               | The Validation value code for the status of the Package.    |
| PKG    | WORKBENCH_PACKAGE_NO_LINK | The URL of the Package in the Kintana Workbench.            |
| PKG    | WORKBENCH_PACKAGE_URL     | The URL of the Package screen in the Kintana Workbench.     |
| PKG    | WORKFLOW_ID               | The ID of the Workflow used by the Package.                 |
| PKG    | WORKFLOW_NAME             | The name of the Workflow used by the Package.               |

Table B-12. Package Lines

| Prefix | Token                     | Description   |
|--------|---------------------------|---|
| PKGL   | APP_CODE                  | The App Code for the Package Line.  |
| PKGL   | APP_NAME                  | The name of the Application for the Package Line.   |
| PKGL   | ID                        | The ID of the Package Line in the table KDLV_PACKAGE_LINES.   |
| PKGL   | OBJECT_CATEGORY_CODE      | The Validation value code of the Object Type category of the line.                                    |
| PKGL   | OBJECT_CATEGORY_NAME      | The Validation value meaning of the Object Type category of the line.                                 |
| PKGL   | OBJECT_NAME               | The Object name of the Package Line.  |
| PKG    | OBJECT_REVISION           | The value of the Object Revision column (if any) as specified by the Object Type of the Package Line. |
| PKGL   | OBJECT_TYPE               | The Object Type of the Package Line.  |
| PKGL   | OBJECT_TYPE_ID            | The ID of the Object Type of the Package Line.  |
| PKGL   | PACKAGE_LINE_ID           | The ID of the Package Line.   |
| PKGL   | SEQ                       | The sequence of the Package Line (relative to other lines in the same Package).                       |
| PKGL   | WORKBENCH_OBJECT_TYPE_URL | URL to access the Object Type window for this Object Type in the Kintana Workbench.                   |

Table B-13. Package Pending

| Prefix   | Tokens                 | Description   |
|----------|------------------------|---|
| PKG.PEND | ID                     | The ID of the entity that is being blocked by the Package.  |
| PKG.PEND | NAME                   | The name of the entity that is being blocked by the Package.  |
| PKG.PEND | DETAIL                 | Detail information for the entity that is being blocked by the Package.   |
| PKG.PEND | DESCRIPTION            | The description of the entity that is being blocked by the Package.   |
| PKG.PEND | STATUS_ID              | The ID of the state or code of the status of the entity that is being blocked by the Package.                       |
| PKG.PEND | STATUS_NAME            | The name of the status (or state) of the entity that is being blocked by the Package.                               |
| PKG.PEND | STATE                  | The name of the state of the entity of the Request that is being blocked by the Package.                            |
| PKG.PEND | ASSIGNED_TO_USERNAME   | The name of the assigned user (or resource) of the entity that is being blocked by the Package.                     |
| PKG.PEND | ASSIGNED_TO_USER_ID    | The username of the assigned user (or resource) of the entity that is being blocked by the Package.                 |
| PKG.PEND | ASSIGNED_TO_GROUP_NAME | The name of the assigned group (or resource group) of the entity that is being blocked by the Package.              |
| PKG.PEND | ASSIGNED_TO_GROUP_ID   | The ID of the assigned group (or resource group) of the entity that is being blocked by the Package.                |
| PKG.PEND | RESOURCE_USERNAME      | The name of the resource associated with the entity that is being blocked by the Package.                           |
| PKG.PEND | RESOURCE_ID            | The username of the assigned user (or resource) associated with the entity that is being blocked by the Package.    |
| PKG.PEND | RESOURCE_GROUP_NAME    | The name of the assigned group (or resource group) associated with the entity that is being blocked by the Package. |
| PKG.PEND | RESOURCE_GROUP_ID      | The ID of the assigned group (or resource group) associated with the entity that is being blocked by the Package.   |
| PKG.PEND | PERCENT_COMPLETE       | The current percent complete value associated with the entity that is being blocked by the Package.                 |
| PKG.PEND | ENTITY_TYPE_ID         | The ID of the type of entity that is being blocked by the Package.  |

Table B-13. Package Pending

| Prefix   | Tokens           | Description  |
|----------|------------------|--|
| PKG.PEND | ENTITY_TYPE_NAME | The name of the type of entity that is being blocked by the Package. |

Table B-14. Program

| Prefix | Token                    | Description   |
|--------|--------------------------|---|
| PRG    | CREATED_BY               | The ID of the user that created the Program.              |
| PRG    | CREATED_BY_USERNAME      | The name of the user that created the Program.            |
| PRG    | LAST_UPDATED_BY          | The ID of the user that last updated the Program.         |
| PRG    | LAST_UPDATED_BY_USERNAME | The name of the user that last updated the Program.       |
| PRG    | PROGRAM_MANAGER          | The ID(s) of the user(s) assigned to manage this Program. |

Table B-15. Projects

| Prefix | Tokens              | Description  |
|--------|---------------------|--|
| PRJ    | ACTUAL_DURATION     | The actual duration of the Project.                    |
| PRJ    | ACTUAL_EFFORT       | The actual effort associated with the Project.         |
| PRJ    | ACTUAL_FINISH_DATE  | The actual finish date of the Project.                 |
| PRJ    | ACTUAL_START_DATE   | The actual start date of the Project.                  |
| PRJ    | BUDGET_ID           | The ID of the Budget linked to the Project.            |
| PRJ    | BUDGET_NAME         | The name of the Budget linked to the Project.          |
| PRJ    | CONFIDENCE_CODE     | The code of the confidence value entered by the user.  |
| PRJ    | CONFIDENCE_NAME     | The name of the confidence value entered by the user.  |
| PRJ    | CREATED_BY          | The user who created the Project.                      |
| PRJ    | CREATED_BY_EMAIL    | The email address of the user who created the Project. |
| PRJ    | CREATED_BY_USERNAME | The username of the person who created the Project.    |
| PRJ    | CREATION_DATE       | The creation date of the Project.                      |
| PRJ    | DEPARTMENT_CODE     | The code of the department value entered by the user.  |
| PRJ    | DEPARTMENT_NAME     | The name of the department value entered by the user.  |
| PRJ    | DESCRIPTION         | The description of the Project.                        |

Table B-15. Projects

| Prefix | Tokens                       | Description  |
|--------|------------------------------|--|
| PRJ    | ESTIMATED_REMAINING_DURATION | The estimated remaining duration of the Project.   |
| PRJ    | ESTIMATED_REMAINING EffORT   | The estimated remaining effort involved in the Project.  |
| PRJ    | ESTIMATED_FINISH_DATE        | The estimated finish date of the Project.  |
| PRJ    | LAST_UPDATE_DATE             | The date the Project was last updated.   |
| PRJ    | LAST_UPDATED_BY              | The last person to update the Project.   |
| PRJ    | LAST_UPDATED_BY_EMAIL        | The email address of the last person to update the Project.  |
| PRJ    | LAST_UPDATED_BY_USERNAME     | The username of the last person to update the Project.   |
| PRJ    | MASTER_PROJECT_ID            | The ID of the Master Project.  |
| PRJ    | MASTER_PROJECT_NAME          | The name of the Master Project.  |
| PRJ    | PARENT_PROJECT_ID            | The ID of the parent Project.  |
| PRJ    | PARENT_PROJECT_NAME          | The name of the parent Project.  |
| PRJ    | PERCENT_COMPLETE             | The Project's completed percentage.  |
| PRJ    | PRIORITY                     | The priority of the Project.   |
| PRJ    | PROJECT_ID                   | The number that uniquely identifies the Project (same as PROJECT_NUMBER) in the table KDRV_PROJECTS. |
| PRJ    | PROJECT_MANAGER              | The Manager of the Project.  |
| PRJ    | PROJECT_MANAGER_EMAIL        | The email address of the Project Manager.  |
| PRJ    | PROJECT_MANAGER_USERNAME     | The username of the Project Manager.   |
| PRJ    | PROJECT_NAME                 | The name of the Project.   |
| PRJ    | PROJECT_NAME_LINK            | Shows up as a standard hyperlink to the Project in HTML-format Notifications.                        |
| PRJ    | PROJECT_NUMBER               | The number that uniquely identifies the Project (same as PROJECT_ID).                                |
| PRJ    | PROJECT_PATH                 | The Project Path. This is a hierarchy of parent Projects that contain this Project.                  |
| PRJ    | PROJECT_STATE                | The Project State.   |
| PRJ    | PROJECT_TEMPLATE             | The name of the Project Template used to create the Project.   |
| PRJ    | PROJECT_TYPE_CODE            | Returns TASK for Tasks and PROJECT for Projects.   |
| PRJ    | PROJECT_URL                  | The URL for the Project Overview.  |
| PRJ    | SCHEDULED EffORT             | The scheduled effort defined in the Project.   |

Table B-15. Projects

| Prefix | Tokens                | Description                                      |
|--------|-----------------------|--|
| PRJ    | SCHEDULED_DURATION    | The Project's scheduled duration.                |
| PRJ    | SCHEDULED_FINISH_DATE | The Project's scheduled finish date.             |
| PRJ    | SCHEDULED_START_DATE  | The Project's scheduled start date.              |
| PRJ    | SUMMARY_CONDITION     | The Project's Summary Condition.                 |
| PRJ    | WORKBENCH_PROJECT_URL | The URL to access this Project in the Workbench. |

Table B-16. Project Details

| Prefix | Tokens*           | Description   |
|--------|-------------------|---|
| PRJD   | PROJECT_DETAIL_ID | The ID of Project Detail in the table KDRV_PROJECT_DETAILS. |
| PRJD   | PROJECT_ID        | The ID of the Project in the table KDRV_PROJECT_DETAILS.    |

\* Parameters are accessible with this prefix (similar to Request Detail):  
[PRJD.P.CUSOM\_TOKEN].

Table B-17. Releases

| Prefix | Tokens                       | Description  |
|--------|------------------------------|--|
| REL    | RELEASE_ID                   | The ID of the Release in the table KREL_RELEASES.              |
| REL    | RELEASE_NAME                 | The name of the Release.                                       |
| REL    | RELEASE_STATUS               | The status of the Release.                                     |
| REL    | CREATED_BY                   | The ID of the user who created the Release.                    |
| REL    | CREATED_BY_USERNAME          | The Kintana username of the user who created the Release.      |
| REL    | LAST_UPDATED_BY              | The ID of the user who last updated the Release.               |
| REL    | LAST_UPDATED_BY_USER<br>NAME | The Kintana username of the user who last updated the Release. |
| REL    | LAST_UPDATE_DATE             | The date that the Release was last updated.                    |



Table B-17. Releases

| Prefix | Tokens          | Description   |
|--------|-----------------|---|
| REL    | RELEASE_MANAGER | The Kintana user who is designated the Release Manager. |
| REL    | RELEASE_TEAM    | The group of Kintana users associated with the Release. |
| REL    | RELEASE_GROUP   | The high level categorization of the Release.           |
| REL    | DESCRIPTION     | The description of the Release.                         |
| REL    | NOTES           | The notes contained within the Release.                 |

Table B-18. Requests

| Prefix | Token                  | Description  |
|--------|------------------------|--|
| REQ    | APPLICATION_CODE       | The Validation value code for the application that the Request is assigned to.   |
| REQ    | APPLICATION_NAME       | The Validation value meaning of the application that the Request is assigned to. |
| REQ    | ASSIGNED_TO_EMAIL      | The email address of the user the Request has been assigned to.                  |
| REQ    | ASSIGNED_TO_GROUP_ID   | The ID of the Security Group that the Request has been assigned to.              |
| REQ    | ASSIGNED_TO_GROUP_NAME | The name of the Security Group that the Request has been assigned to.            |
| REQ    | ASSIGNED_TO_USERNAME   | The Kintana username of the user that the Request has been assigned to.          |
| REQ    | ASSIGNED_TO_USER_ID    | The ID of the user that the Request has been assigned to.                        |
| REQ    | COMPANY                | The Company employing the user that created the Request.                         |
| REQ    | COMPANY_NAME           | The name of the Company employing the user that created the Request.             |
| REQ    | CONTACT_EMAIL          | The email address of the Contact for the Request.                                |
| REQ    | CONTACT_NAME           | The full name of the Contact for the Request.                                    |
| REQ    | CONTACT_PHONE_NUMBER   | The phone number of the Contact for the Request.                                 |
| REQ    | CREATED_BY             | The ID of the user that created the Request.                                     |
| REQ    | CREATED_BY_EMAIL       | The email address of the user that created the Request.                          |
| REQ    | CREATED_BY_USERNAME    | The Kintana username of the user that created the Request.                       |
| REQ    | CREATION_DATE          | The date the Request was created.  |
| REQ    | DEPARTMENT_CODE        | The Validation value code of the department for the Request.                     |

Table B-18. Requests

| Prefix | Token                      | Description   |
|--------|----------------------------|---|
| REQ    | DEPARTMENT_NAME            | The Validation value meaning of the department for the Request.               |
| REQ    | DESCRIPTION                | The description of the Request.   |
| REQ    | LAST_UPDATED_BY            | The ID of the user that last updated the Request.                             |
| REQ    | LAST_UPDATED_BY_EMAIL      | The email address of the user that last updated the Request.                  |
| REQ    | LAST_UPDATED_BY_USERNAME   | The Kintana username of the user that last updated the Request.               |
| REQ    | LAST_UPDATE_DATE           | The date the Request was last updated.  |
| REQ    | NOTES                      | The notes for the Request.  |
| REQ    | PERCENT_COMPLETE           | The percent complete of the Request.  |
| REQ    | PRIORITY_CODE              | The Validation value code of the Request priority.                            |
| REQ    | PRIORITY_NAME              | The Validation value meaning of the Request priority.                         |
| REQ    | PROJECT_CODE               | The Validation value code of the Project the Request belongs to.              |
| REQ    | PROJECT_NAME               | The Validation value meaning of the Project the Request belongs to.           |
| REQ    | SUBMIT_DATE                | The date that the Request was submitted.                                      |
| REQ    | REQUEST_GROUP_CODE         | The code for the Request Group.   |
| REQ    | REQUEST_GROUP_NAME         | The name of the Request Group.  |
| REQ    | REQUEST_ID                 | The ID of the Request in the table KCRT_REQUESTS.                             |
| REQ    | REQUEST_ID_LINK            | Shows up as a standard hyperlink to the Request in HTML-format Notifications. |
| REQ    | REQUEST_SUB_TYPE_ID        | The ID of the sub-type for the Request.                                       |
| REQ    | REQUEST_SUB_TYPE_NAME      | The name of the sub-type for the Request.                                     |
| REQ    | REQUEST_TYPE_ID            | The ID of the Request Type of the Request.                                    |
| REQ    | REQUEST_TYPE_NAME          | The name of the Request Type of the Request.                                  |
| REQ    | REQUEST_URL                | URL of the Request in Kintana HTML.   |
| REQ    | STATUS_ID                  | The ID of the status of the Request.  |
| REQ    | STATUS_NAME                | The status of the Request.  |
| REQ    | WORKBENCH_REQUEST_TYPE_URL | The URL of the Request Type in the Kintana Workbench.                         |
| REQ    | WORKFLOW_ID                | The ID of the Workflow used by the Request.                                   |
| REQ    | WORKFLOW_NAME              | The name of the Workflow used by the Request.                                 |

Figure B-1 Request Details

| Prefix* | Tokens            | Description  |
|---------|-------------------|--|
| REQD    | CREATED_BY        | The ID of the User who created the Request Detail.               |
| REQD    | CREATION_DATE     | The date the Request Detail was created.                         |
| REQD    | LAST_UPDATED_BY   | The ID of the User that last updated the Request Detail.         |
| REQD    | LAST_UPDATE_DATE  | The date the Request Detail was last updated.                    |
| REQD    | REQUEST_DETAIL_ID | The ID for the Request Detail in the table KCRT_REQUEST_DETAILS. |
| REQD    | REQUEST_ID        | The ID of the Request for the Request Detail.                    |
| REQD    | REQUEST_TYPE_ID   | The ID of the Request Type for the Request Detail.               |

\* Prefix is mainly used for accessing custom fields: [REQD.P.CUSTOM\_TOKEN]

Table B-19. Request Pending

| Prefix   | Tokens                 | Description  |
|----------|------------------------|--|
| REQ.PEND | ID                     | The ID of the entity that is being blocked by the Request.   |
| REQ.PEND | NAME                   | The name of the entity that is being blocked by the Request.   |
| REQ.PEND | DETAIL                 | Detail information for the entity that is being blocked by the Request.                                |
| REQ.PEND | DESCRIPTION            | The description of the entity that is being blocked by the Request.                                    |
| REQ.PEND | STATUS_ID              | The ID of the state or code of the status of the entity that is being blocked by the Request.          |
| REQ.PEND | STATUS_NAME            | The name of the status (or state) of the entity that is being blocked by the Request.                  |
| REQ.PEND | STATE                  | The name of the state of the entity of the Request that is being blocked by the Request.               |
| REQ.PEND | ASSIGNED_TO_USERNAME   | The name of the assigned user (or resource) of the entity that is being blocked by the Request.        |
| REQ.PEND | ASSIGNED_TO_USER_ID    | The username of the assigned user (or resource) of the entity that is being blocked by the Request.    |
| REQ.PEND | ASSIGNED_TO_GROUP_NAME | The name of the assigned group (or resource group) of the entity that is being blocked by the Request. |
| REQ.PEND | ASSIGNED_TO_GROUP_ID   | The ID of the assigned group (or resource group) of the entity that is being blocked by the Request.   |

Table B-19. Request Pending

| Prefix   | Tokens              | Description   |
|----------|---------------------|---|
| REQ.PEND | RESOURCE_USERNAME   | The name of the resource associated with the entity that is being blocked by the Request.                           |
| REQ.PEND | RESOURCE_ID         | The username of the assigned user (or resource) associated with the entity that is being blocked by the Request.    |
| REQ.PEND | RESOURCE_GROUP_NAME | The name of the assigned group (or resource group) associated with the entity that is being blocked by the Request. |
| REQ.PEND | RESOURCE_GROUP_ID   | The ID of the assigned group (or resource group) associated with the entity that is being blocked by the Request.   |
| REQ.PEND | PERCENT_COMPLETE    | The current percent complete value associated with the entity that is being blocked by the Request.                 |
| REQ.PEND | ENTITY_TYPE_ID      | The ID of the type of entity that is being blocked by the Request.  |
| REQ.PEND | ENTITY_TYPE_NAME    | The name of the type of entity that is being blocked by the Request.  |

Table B-20. Report Submissions

| Prefix | Tokens               | Description   |
|--------|----------------------|---|
| RP     | CREATED_BY           | The ID of the User who submitted the Report.                            |
| RP     | CREATION_DATE        | The date the Report was submitted.                                      |
| RP     | FILENAME             | The filename for the Report. This file name is found in the REPORT_URL. |
| RP     | LAST_UPDATED_BY      | The ID of the User that last updated the Report submission.             |
| RP     | LAST_UPDATE_DATE     | The date the Report submission was last updated.                        |
| RP     | NEW_STATUS           | The visible value for the Report's new Status.                          |
| RP     | NEW_STATUS_CODE      | The code for the Report's new Status.                                   |
| RP     | OLD_STATUS           | The visible value for the Report's old status.                          |
| RP     | OLD_STATUS_CODE      | The code for the Report's old status.                                   |
| RP     | REPORT_LOG_URL       | The Web address where the Report log is located.                        |
| RP     | REPORT_SUBMISSION_ID | The ID of the Report submission in the table KNTA_REPORT_SUBMISSIONS.   |

Table B-20. Report Submissions

|    |                           |  |
|----|---------------------------|--|
| RP | REPORT_TYPE_NAME          | The name of the Report Type of the Report submission.              |
| RP | REPORT_TYPE_ID            | The ID of the Report Type of the Report submission.                |
| RP | REPORT_URL                | The Web address where the Report output is located.                |
| RP | STATUS                    | The status of the Report submission.                               |
| RP | STATUS_CODE               | The Validation value code for the status of the Report submission. |
| RP | WORKBENCH_REPORT_TYPE_URL | The URL of the Report Type in the Kintana Workbench.               |

Table B-21. Resource Pools

| Prefix | Tokens                      | Description   |
|--------|-----------------------------|---|
| RSCP   | ACTIVE_FLAG                 | The active flag of the Resource Pool.                                       |
| RSCP   | CREATED_BY                  | The username of the user who created the Resource Pool.                     |
| RSCP   | CREATION_DATE               | The date that the Resource Pool was created.                                |
| RSCP   | DESCRIPTION                 | The description of the Resource Pool.                                       |
| RSCP   | END_PERIOD                  | The end period of the Resource Pool.  |
| RSCP   | INITIATION_REQ              | The initiation Request ID of the Resource Pool.                             |
| RSCP   | PERIOD_SIZE                 | The period size of the Resource Pool.                                       |
| RSCP   | RESOURCE_POOL_URL           | The URL to view this Resource Pool.   |
| RSCP   | RSC_POOL_ID                 | The ID of the Resource Pool in table KRSC_RSC_POOLS.                        |
| RSCP   | RSC_POOL_IS_FOR_ENTITY_NAME | The entity name to which the Resource Pool is linked (Program or Org Unit). |
| RSCP   | RSC_POOL_IS_FOR_ID          | The ID of the Program or Org unit to which the Resource Pool is linked.     |
| RSCP   | RSC_POOL_IS_FOR_NAME        | The name of the Program or ORg unit to which the Resource Pool is linked.   |
| RSCP   | RSC_POOL_NAME               | The name of the Resource Pool.  |
| RSCP   | START_PERIOD                | The start period of the Resource Pool.                                      |
| RSCP   | STATUS_CODE                 | The status code of the Resource Pool.                                       |

*Table B-21. Resource Pools*

| <b>Prefix</b> | <b>Tokens</b> | <b>Description</b>                    |
|---------------|---------------|---------------------------------------|
| RSCP          | STATUS_NAME   | The status name of the Resource Pool. |

*Table B-22. Security Groups*

| <b>Prefix</b> | <b>Tokens</b>       | <b>Description</b>  |
|---------------|---------------------|---|
| SG            | CREATED_BY          | The ID of the User who created the Security Group.              |
| SG            | CREATION_DATE       | The date the Security Group was created.                        |
| SG            | DESCRIPTION         | The description for the Security Group.                         |
| SG            | LAST_UPDATED_BY     | The ID of the User that last updated the Security Group.        |
| SG            | LAST_UPDATE_DATE    | The date the Security Group was last updated.                   |
| SG            | SECURITY_GROUP_ID   | The ID of the Security Group in the table KNTA_SECURITY_GROUPS. |
| SG            | SECURITY_GROUP_NAME | The name of the Security Group.                                 |

*Table B-23. Skill*

| <b>Prefix</b> | <b>Tokens</b>       | <b>Description</b>  |
|---------------|---------------------|---|
| SKL           | AVERAGE_COST_RATE   | The average cost rate associated with the skill.                        |
| SKL           | CREATED_BY          | The user ID that created the Skill.                                     |
| SKL           | CREATED_BY_USERNAME | The name of the user that created the Skill.                            |
| SKL           | CREATION_DATE       | The date that the Skill was created.                                    |
| SKL           | SKILL_CATEGORY_CODE | The lookup code of Skill Category (lookup type = RSC - Skill Category). |
| SKL           | SKILL_CATEGORY_NAME | The name of the Skill category.   |
| SKL           | SKILL_ID            | The ID of the Skill in table KRSC_SKILLS.                               |
| SKL           | SKILL_NAME          | The name of the Skill.  |

Table B-24. Staffing Profile

| Prefix | Tokens                        | Description   |
|--------|-------------------------------|---|
| STFP   | ACTIVE_FLAG                   | The active flag of the Staffing Profile.  |
| STFP   | CREATED_BY                    | The username of the user who created the Staffing Profile.  |
| STFP   | CREATION_DATE                 | The date that the Staffing Profile was created.   |
| STFP   | DESCRIPTION                   | The description of the Staffing Profile.  |
| STFP   | END_PERIOD                    | The end period of the Staffing Profile.   |
| STFP   | INITIATION_REQ                | The initiation Request ID of the Staffing Profile.  |
| STFP   | PERIOD_SIZE                   | The period size of the Staffing Profile.  |
| STFP   | STAFFING_PROFILE_URL          | The URL to view this Staffing Profile.  |
| STFP   | STAFF_PROF_ID                 | The ID of the Staffing Profile in table KRSC_STAFF_PROFS.   |
| STFP   | STAFF_PROF_IS_FOR_ENTITY_NAME | The entity name to which the Staffing Profile is linked.  |
| STFP   | STAFF_PROF_IS_FOR_ID          | The ID of the Project, Program or Org unit to which the Staffing Profile is linked.                                   |
| STFP   | STAFF_PROFL_IS_FOR_NAME       | The name of the Project, Program or Org unit to which the Staffing Profile is linked (Project, Program, or Org Unit). |
| STFP   | STAFF_PROF_NAME               | The name of the Staffing Profile.   |
| STFP   | START_PERIOD                  | The start period of the Staffing Profile.   |
| STFP   | STATUS_CODE                   | The status code of the Staffing Profile.  |
| STFP   | STATUS_NAME                   | The status name of the Staffing Profile.  |

Table B-25. System

| Prefix | Tokens            | Description  |
|--------|-------------------|--|
| SYS    | DATE              | The date at the time the token is parsed.  |
| SYS    | NEWLINE           | A new line character.  |
| SYS    | TIME_STAMP        | A date and time stamp at the time the token is parsed.   |
| SYS    | UNIQUE_IDENTIFIER | Used to obtain a unique number from the database. It can be used to generate unique filenames, etc. It is often necessary to use with the 'ksc_set' Special Command. |
| SYS    | UNIX_NEWLINE      | The UNIX new line character.   |

Table B-25. System

|     |          |   |
|-----|----------|---|
| SYS | USERNAME | The Kintana username of the User currently logged onto Kintana. |
| SYS | USER_ID  | The ID of the User currently logged onto Kintana.               |

Table B-26. Tasks

| Prefix | Tokens                       | Description  |
|--------|------------------------------|--|
| TSK    | ACTUAL_DURATION              | The actual duration of the Task.                         |
| TSK    | ACTUAL_EFFORT                | The actual effort associated with the Task.              |
| TSK    | ACTUAL_FINISH_DATE           | The actual finish date of the Task.                      |
| TSK    | ACTUAL_START_DATE            | The actual start date of the Task.                       |
| TSK    | CONFIDENCE_CODE              | The code of the confidence value entered by the user.    |
| TSK    | CONFIDENCE_NAME              | The name of the confidence value entered by the user.    |
| TSK    | CONSTRAINT_DATE              | The Task's constraint date.                              |
| TSK    | CREATED_BY                   | The user who created the Task.                           |
| TSK    | CREATED_BY_EMAIL             | The email address of the user who created the Task.      |
| TSK    | CREATED_BY_USERNAME          | The username of the person who created the Task.         |
| TSK    | CREATION_DATE                | The creation date of the Task.                           |
| TSK    | DEPARTMENT_CODE              | The code of the department value entered by the user.    |
| TSK    | DEPARTMENT_NAME              | The name of the department value entered by the user.    |
| TSK    | DESCRIPTION                  | The description of the Task.                             |
| TSK    | ESTIMATED_REMAINING_DURATION | The estimated remaining duration of the Task.            |
| TSK    | ESTIMATED_REMAINING_EFFORT   | The estimated remaining effort involved in the Task.     |
| TSK    | ESTIMATED_FINISH_DATE        | The estimated finish date of the Task.                   |
| TSK    | HAS_EXCEPTIONS               | The flag to show whether or not the Task has exceptions. |
| TSK    | LAST_UPDATE_DATE             | The date the Task was last updated.                      |
| TSK    | LAST_UPDATED_BY              | The last person to update the Task.                      |
| TSK    | LAST_UPDATED_BY_EMAIL        | The email address of the last person to update the Task. |
| TSK    | LAST_UPDATED_BY_USERNAME     | The username of the last person to update the Task.      |
| TSK    | MASTER_PROJECT_ID            | The ID of the Master Project.                            |
| TSK    | MASTER_PROJECT_NAME          | The name of the Master Project.                          |



Table B-26. Tasks

| Prefix | Tokens                | Description   |
|--------|-----------------------|---|
| TSK    | PARENT_PROJECT_ID     | The ID of the parent Project.   |
| TSK    | PARENT_PROJECT_NAME   | The name of the parent Project.   |
| TSK    | PERCENT_COMPLETE      | The Task's completed percentage.  |
| TSK    | PRIORITY              | The priority of the Task.   |
| TSK    | PROJECT_PATH          | The Project Path. Hierarchy of parent Projects that contain this Task.  |
| TSK    | PROJECT_TEMPLATE      | The name of the Project Template used to create the Project containing the Task.  |
| TSK    | PROJECT_TYPE_CODE     | Returns TASK for Tasks and PROJECT for Projects.  |
| TSK    | RESOURCE_ID           | The ID of the Resource assigned to the Task.  |
| TSK    | RESOURCE_EMAIL        | The email address of the Resource.  |
| TSK    | RESOURCE_GROUP_ID     | The ID of the Resource Group assigned to the Task.  |
| TASK   | RESOURCE_GROUP_NAME   | The name of the Resource Group assigned to the Task.  |
| TSK    | RESOURCE_USERNAME     | The username of the Resource.   |
| TSK    | SCHEDULED_EFFORT      | The scheduled effort involved in the Task.  |
| TSK    | SCHEDULED_DURATION    | The Task's scheduled duration.  |
| TSK    | SCHEDULED_FINISH_DATE | The Task's scheduled finish date.   |
| TSK    | SCHEDULED_START_DATE  | The Task's scheduled start date.  |
| TSK    | SCHEDULING CONSTRAINT | The Task's scheduling constraint.   |
| TSK    | TASK_CATEGORY         | The predefined category the Task belongs to.  |
| TSK    | TASK_ID               | The number that uniquely identifies the Task (same as TASK_NUMBER). This corresponds to the PROJECT_ID column in table KDRV_PROJECTS. |
| TSK    | TASK_NAME             | The name of the Task.   |
| TSK    | TASK_NAME_LINK        | Standard hyperlink to the Task in HTML-format Notifications.  |
| TSK    | TASK_NUMBER           | The number that uniquely identifies the Task (same as TASK_ID).   |
| TSK    | TASK_STATE            | The Task State.   |
| TSK    | TASK_URL              | The URL for the Task Detail page.   |
| TSK    | WORKBENCH_TASK_URL    | The URL to access this Task in the Workbench.   |

Table B-27. Tasks Pending

| Prefix   | Tokens                 | Description   |
|----------|------------------------|---|
| TSK.PEND | ID                     | The ID of the entity that is being blocked by the Task.   |
| TSK.PEND | NAME                   | The name of the entity that is being blocked by the Task.   |
| TSK.PEND | DETAIL                 | Detail information for the entity that is being blocked by the Task as shown in the References panel.           |
| TSK.PEND | DESCRIPTION            | The description of the entity that is being blocked by the Task.  |
| TSK.PEND | STATUS_ID              | The ID of the state or the code of the status of the entity that is being blocked by the Task.                  |
| TSK.PEND | STATUS_NAME            | The name of the status (or state) of the entity that is being blocked by the Task.                              |
| TSK.PEND | STATE                  | The name of the state of the entity that is being blocked by the Task.  |
| TSK.PEND | ASSIGNED_TO_USERNAME   | The name of the assigned user (or resource) of the entity that is being blocked by the Task.                    |
| TSK.PEND | ASSIGNED_TO_USER_ID    | The username of the assigned user (or resource) of the entity that is being blocked by the Task.                |
| TSK.PEND | ASSIGNED_TO_GROUP_NAME | The name of the assigned group (or resource group) of the entity that is being blocked by the Task.             |
| TSK.PEND | ASSIGNED_TO_GROUP_ID   | The ID of the assigned group (or resource group) of the entity that is being blocked by the Task.               |
| TSK.PEND | RESOURCE_USERNAME      | The name of the resource associated with the entity that is being blocked by the Task.                          |
| TSK.PEND | RESOURCE_ID            | The username of the resource (or assigned user) associated with the entity that is being blocked by the Task.   |
| TSK.PEND | RESOURCE_GROUP_NAME    | The name of the resource group (or assigned user) associated with the entity that is being blocked by the Task. |
| TSK.PEND | RESOURCE_GROUP_ID      | The ID of the resource group (or assigned group) associated with the entity that is being blocked by the Task.  |
| TSK.PEND | PERCENT_COMPLETE       | The current percent complete value associated with the entity that is being blocked by the Task.                |
| TSK.PEND | ENTITY_TYPE_ID         | The ID of the type of entity that is being blocked by the Task.   |

Table B-27. Tasks Pending

| Prefix   | Tokens           | Description   |
|----------|------------------|---|
| TSK.PEND | ENTITY_TYPE_NAME | The name of the type of entity that is being blocked by the Task. |

Table B-28. Users

| Prefix | Tokens                   | Description   |
|--------|--------------------------|---|
| USR    | AUTHENTICATION_MODE_CODE | The authentication mode for the user. Example: LDAP, Kintana.                           |
| USR    | AUTHENTICATION_MODE_NAME | The authentication mode for the user. Example: LDAP, Kintana.                           |
| USR    | COMPANY                  | The Company employing the user.   |
| USR    | COMPANY_NAME             | The name of the Company employing the user.   |
| USR    | COST_RATE                | The cost rate of the user (\$/hour - subject to security of user evaluating the Token). |
| USR    | CREATED_BY               | The ID of the user that created the user.   |
| USR    | CREATED_BY_USERNAME      | The Kintana username of the user that created the user.                                 |
| USR    | CREATION_DATE            | The date the user was created.  |
| USR    | DEPARTMENT_CODE          | The lookup code of the department the user belongs to (lookup type = DEPT).             |
| USR    | DEPARTMENT_NAME          | The name of the department that the user belongs to.                                    |
| USR    | EMAIL_ADDRESS            | The email address of the user.  |
| USR    | END_DATE                 | The date the user is made inactive in the application.                                  |
| USR    | FIRST_NAME               | The first name of the user.   |
| USR    | LAST_NAME                | The last name of the user.  |
| USR    | LAST_UPDATED_BY          | The ID of the user that last updated the user.  |
| USR    | LAST_UPDATED_BY_USERNAME | The Kintana username of the user that last updated the user.                            |
| USR    | LAST_UPDATE_DATE         | The date the user was last updated.   |
| USR    | LOCATION_CODE            | The lookup code of the user's location (lookup type = RSC - Location).                  |
| USR    | LOCATION_NAME            | The name of the user's location.  |
| USR    | MANAGER_USERNAME         | The username of the user's manager.   |
| USR    | MANAGER_USER_ID          | The ID of the user's manager.   |
| USR    | PASSWORD                 | The password for the user to use to log onto Kintana. This value is encrypted.          |
| USR    | PASSWORD_EXPIRATION_DATE | The date the password needs to be reset for the user.                                   |

Table B-28. Users

|     |                          |  |
|-----|--------------------------|--|
| USR | PASSWORD_EXPIRATION_DAYS | The number of days until the password must be reset for the user.                              |
| USR | PHONE_NUMBER             | The phone number of the user.  |
| USR | PRIMARY_SKILL_ID         | The ID of the primary skill associated with the user.  |
| USR | PRIMARY_SKILL_NAME       | The name of the primary skill associated with the user.  |
| USR | RESOURCE_CATEGORY_CODE   | The lookup code of Resource Category (lookup type = RSC - Category) to which the user belongs. |
| USR | RESOURCE_CATEGORY_NAME   | The name of the category to which the user belongs.  |
| USR | RESOURCE_TITLE_CODE      | the lookup code of the user's Resource Title (lookup type = RSC - Resource Title).             |
| USR | RESOURCE_TITLE_NAME      | The name of the user's resource title.   |
| USR | START_DATE               | The date the user is made active in the application.   |
| USR | USERNAME                 | The username for the user to use to log onto Kintana.  |
| USR | USER_ID                  | The ID of the user in the table KNTA_USERS.  |
| USR | WORKLOAD_CAPACITY        | The workload capacity of the user (% of FTE)   |

Table B-29. Validations

| Prefix | Tokens                   | Description   |
|--------|--------------------------|---|
| VAL    | COMPONENT_TYPE           | The component type associated with the Validation.                |
| VAL    | CREATED_BY               | The ID of the User that created the Validation.                   |
| VAL    | CREATION_DATE            | The date the Validation was created.                              |
| VAL    | DESCRIPTION              | The description of the Validation.                                |
| VAL    | LAST_UPDATED_BY          | The ID of the user that last updated the Validation.              |
| VAL    | LAST_UPDATE_DATE         | The date the Validation was last updated.                         |
| VAL    | LOOKUP_TYPE              | The lookup type associated with the Validation (if applicable).   |
| VAL    | VALIDATION_ID            | The ID of the Validation in the table KNTA_VALIDATIONS.           |
| VAL    | VALIDATION_NAME          | The name of the Validation.                                       |
| VAL    | VALIDATION_SQL           | The SQL statement associated with the Validation (if applicable). |
| VAL    | WORKBENCH_VALIDATION_URL | The URL for the Validation in the Kintana Workbench.              |

Table B-30. Validation Values

| Prefix | Tokens           | Description  |
|--------|------------------|--|
| VALUE  | CREATED_BY       | The ID of the user that created the value.   |
| VALUE  | CREATION_DATE    | The date the value was created.  |
| VALUE  | DEFAULT_FLAG     | The flag to indicate whether the value is the default value for the associated lookup type.                                    |
| VALUE  | DESCRIPTION      | The description of the value.  |
| VALUE  | ENABLED_FLAG     | The flag to indicate whether the value is enabled for selection in a drop-down list.   |
| VALUE  | LAST_UPDATED_BY  | The ID of the user that last updated the value.  |
| VALUE  | LAST_UPDATE_DATE | The date the value was last updated.   |
| VALUE  | LOOKUP_CODE      | The code associated with the value.  |
| VALUE  | LOOKUP_TYPE      | The lookup type the value belongs to.  |
| VALUE  | MEANING          | The meaning associated with the value.   |
| VALUE  | SEQ              | The sequence relative to other values in the associated lookup type in which this value will be displayed in a drop-down list. |

Table B-31. Workflows

| Prefix | Tokens                    | Description   |
|--------|---------------------------|---|
| WF     | CREATED_BY                | The ID of the User that created the Workflow.   |
| WF     | CREATION_DATE             | The date the Workflow was created.  |
| WF     | DESCRIPTION               | The description of the Workflow.  |
| WF     | ENABLED_FLAG              | The flag indicating whether the Workflow is enabled and available to use in Packages and/or Requests. |
| WF     | FIRST_WORKFLOW_STEP_ID    | The ID of the first Workflow Step in the Workflow.  |
| WF     | FIRST_WORKFLOW_STEP_NAME  | The name of the first Workflow Step in the Workflow.  |
| WF     | ICON_NAME                 | The name of the Workflow Step icon.   |
| WF     | LAST_UPDATED_BY           | The ID of the user that last updated the Workflow.  |
| WF     | LAST_UPDATE_DATE          | The date the Workflow was last updated.   |
| WF     | PRODUCT_SCOPE_CODE        | The Validation value code for the product scope of the Workflow.                                      |
| WF     | REOPEN_WORKFLOW_STEP_ID   | The ID of the reopened workflow step.   |
| WF     | REOPEN_WORKFLOW_STEP_NAME | The name of the reopened workflow step.   |
| WF     | SUBWORKFLOW_FLAG          | An indicator that specifies whether this Workflow can be used as a Subworkflow.                       |
| WF     | WORKFLOW_ID               | The ID of the Workflow defined in the table KWFL_WORKFLOWS.   |

Table B-31. Workflows

|    |                        |  |
|----|------------------------|--|
| WF | WORKFLOW_NAME          | The name of the Workflow.                              |
| WF | WORKBENCH_WORKFLOW_URL | The URL to open the Workflow in the Kintana Workbench. |

Table B-32. Workflow Steps

| Prefix | Tokens                      | Description  |
|--------|-----------------------------|--|
| WFS    | ACTION_BUTTON_LABEL         | The label displayed on the Package or Request action button for the Workflow Step.   |
| WFS    | AVERAGE_LEAD_TIME           | The average lead time in days defined for the Workflow Step.   |
| WFS    | CREATED_BY                  | The ID of the user that created the Workflow Step.   |
| WFS    | CREATION_DATE               | The date the Workflow Step was created.  |
| WFS    | DESCRIPTION                 | The description of the Workflow Step.  |
| WFS    | DEST_ENV_GROUP_ID           | The ID of the destination Environment Group for the Workflow Step.   |
| WFS    | DEST_ENV_GROUP_NAME         | The name of the destination Environment Group for the Workflow Step.   |
| WFS    | DEST_ENVIRONMENT_ID         | The ID of destination Environment for the Workflow Step.   |
| WFS    | DEST_ENVIRONMENT_NAME       | The name of the destination Environment for the Workflow Step.   |
| WFS    | ENABLED_FLAG                | The flag indicating whether the Workflow Step is enabled and able to be traversed in a Package or Request.                     |
| WFS    | GL_ARCHIVE_FLAG             | For GL object migration, the flag indicating whether to save the GL object being migrated to the GL*Migrator archive.          |
| WFS    | INFORMATION_URL             | The Workflow Step's information URL.   |
| WFS    | JUMP_RECEIVE_LABEL_CODE     | The code for a Jump/Receive Workflow Step.   |
| WFS    | JUMP_RECEIVE_LABEL_NAME     | The name of a Jump/Receive Workflow Step.  |
| WFS    | LAST_UPDATED_BY             | The ID of the User that last updated the Workflow Step.  |
| WFS    | LAST_UPDATE_DATE            | The date the Workflow Step was last updated.   |
| WFS    | OM_ARCHIVE_FLAG             | For AOL object migration, the flag indicating whether to save the AOL object being migrated to the Object*Migrator archive.    |
| WFS    | PARENT_ASSIGNED_TO_GROUP_ID | The ID of the Security Group that the current Package or Request is assigned to (determined by context at time of evaluation). |

Table B-32. Workflow Steps

|     |                                |  |
|-----|--------------------------------|--|
| WFS | PARENT_ASSIGNED_TO_GROUP_NAME  | The Security Group that the current Package or Request is assigned to (determined by context at time of evaluation).   |
| WFS | PARENT_ASSIGNED_TO_USERNAME    | The name of the user that the current Package or Request is assigned to (determined by context at time of evaluation). |
| WFS | PARENT_ASSIGNED_TO_USER_ID     | The ID of the user that the current Package or Request is assigned to (determined by context at time of evaluation).   |
| WFS | PARENT_STATUS                  | The Validation value code of the status of the Request that is using the Workflow Step.                                |
| WFS | PARENT_STATUS_NAME             | The Validation value meaning of the status of the Request that is using the Workflow Step.                             |
| WFS | PRODUCT_SCOPE_CODE             | The Validation value code for the product scope of the Workflow containing the Workflow Step.                          |
| WFS | RESULT_WORKFLOW_PARAMETER_ID   | The ID of the Workflow parameter that the result of the Workflow Step is written to.                                   |
| WFS | RESULT_WORKFLOW_PARAMETER_NAME | The name of the Workflow parameter that the result of the Workflow Step is written to.                                 |
| WFS | SORT_ORDER                     | The display sequence of the Workflow Step relative to all other Steps in the Workflow.                                 |
| WFS | SOURCE_ENV_GROUP_ID            | The ID of the source Environment Group for the Workflow Step.  |
| WFS | SOURCE_ENV_GROUP_NAME          | The name of the source Environment Group for the Workflow Step.  |
| WFS | SOURCE_ENVIRONMENT_ID          | The ID of the source Environment for the Workflow Step.  |
| WFS | SOURCE_ENVIRONMENT_NAME        | The name of the source Environment for the Workflow Step.  |
| WFS | STEP_NAME                      | The name of the Workflow Step.   |
| WFS | STEP_NO                        | The display sequence of the Workflow Step relative to all other Steps in the Workflow.                                 |
| WFS | STEP_SOURCE_NAME               | The name of the Workflow Step Source.  |
| WFS | STEP_TYPE_NAME                 | The name of the Workflow Step Source type.   |
| WFS | WORKFLOW_ID                    | The ID of the Workflow containing the Workflow Step.   |
| WFS | WORKFLOW_NAME                  | The name of the Workflow containing the Workflow Step.   |
| WFS | WORKFLOW_STEP_ID               | The ID of the Workflow Step in the table KWFL_WORKFLOW_STEPS.  |

Table B-33. Workflow Step Transaction

| Prefix | Tokens                   | Description  |
|--------|--------------------------|--|
| WST    | CONCURRENT_REQUEST_ID    | The ID of the concurrent request that was launched in Oracle Applications. |
| WST    | CREATED_BY               | The ID of the user that created the Step transaction.                      |
| WST    | CREATION_DATE            | The date the Step transaction was created.                                 |
| WST    | ERROR_MESSAGE            | The error message for the Step transaction.                                |
| WST    | EXECUTION_BATCH_ID       | The ID of the Execution Batch for the Workflow Step.                       |
| WST    | HIDDEN_STATUS            | The hidden value for the status of the Step transaction.                   |
| WST    | LAST_UPDATED_BY          | The ID of the user that last updated the Step transaction.                 |
| WST    | LAST_UPDATED_BY_EMAIL    | The email address of the user that last updated the Step transaction.      |
| WST    | LAST_UPDATED_BY_USERNAME | The Kintana username of the user that last updated the Step transaction.   |
| WST    | LAST_UPDATE_DATE         | The date the Step transaction was last updated.                            |
| WST    | NEW_HIDDEN_STATUS        | The new hidden value for the status of the Step transaction.               |
| WST    | NEW_STATUS               | The new status of the Step transaction.                                    |
| WST    | OLD_HIDDEN_STATUS        | The old hidden value for the status of the Step transaction.               |
| WST    | OLD_STATUS               | The old status of the Step transaction.                                    |
| WST    | STATUS                   | The status of the Step transaction.  |
| WST    | STEP_TRANSACTION_ID      | The ID of the Step transaction in the table KWFL_STEP_TRANSACTIONS.        |
| WST    | TIMEOUT_DATE             | The date that the Step transaction times out.                              |
| WST    | USER_COMMENT             | The user comment for the Step transaction.                                 |
| WST    | WORKFLOW_ID              | The ID of the Workflow for the Step transaction.                           |
| WST    | WORKFLOW_STEP_ID         | The ID of the Workflow Step for the Step transaction.                      |

## Field Group Tokens

Field Groups can be attached to Request Header Types to enable additional pre-configured fields on Requests. Field Groups are often delivered as a part of



Kintana Solution functionality. You will only have access to Field Groups associated with products that are licensed at your site.

*Table B-34. Demand Management Field Group Tokens*

| <b>Field</b>           | <b>Token</b>               |
|------------------------|----------------------------|
| SLA Level              | KNTA_SLA_LEVEL             |
| SLA Violation Data     | KNTA_SLA_VIOLATION_DATE    |
| Service Request Date   | KNTA_SLA_SERV_REQUESTED_ON |
| Service Satisfied Date | KNTA_SLA_SERV_SATISFIED_ON |
| Estimated Start Date   | KNTA_EST_START_DATE        |
| Estimated Effort       | KNTA_EFFORT                |
| Reject Date            | KNTA_REJECTED_DATE         |
| Demand Satisfied Date  | KNTA_DEMAND_SATISFIED_DATE |

*Table B-35. Master Project Reference on Request Field Group Tokens*

| <b>Field</b>   | <b>Token</b>         |
|----------------|----------------------|
| Master Project | KNTA_MASTER_PROJ_REF |

*Table B-36. PMO Field Group Tokens*

| <b>Field</b>             | <b>Token</b>           |
|--------------------------|------------------------|
| Escalation Level         | KNTA_ESCALATION_LEVEL  |
| Role Description         | KNTA_ROLE_DESCRIPTION  |
| Risk Impact Level        | KNTA_RISK_IMPACT_LEVEL |
| Probability              | KNTA_PROBABILITY       |
| CR Level                 | KNTA_CR_LEVEL          |
| Business Impact Severity | KNTA_IMPACT_SEVERITY   |

*Table B-37. Program Reference on Request Field Group Tokens*

| <b>Field</b> | <b>Token</b>           |
|--------------|------------------------|
| Program      | KNTA_PROGRAM_REFERENCE |

*Table B-38. Work Item Field Group Tokens*

| <b>Field</b>                           | <b>Token</b>                |
|--|-----------------------------|
| Scheduled Start Date                   | KNTA_USR_SCHED_START_DATE   |
| Actual Start Date                      | KNTA_USR_ACTUAL_START_DATE  |
| Scheduled Finish Date                  | KNTA_USR_SCHED_FINISH_DATE  |
| Actual Finish Date                     | KNTA_USR_ACTUAL_FINISH_DATE |
| Scheduled Duration                     | KNTA_SCHED_DURATION         |
| Actual Duration                        | KNTA_ACTUAL_DURATION        |
| Scheduled Effort                       | KNTA_SCHED_EFFORT           |
| Actual Effort                          | KNTA_ACTUAL_EFFORT          |
| Workload?                              | KNTA_WORKLOAD               |
| Workload Category                      | KNTA_WORKLOAD_CATEGORY      |
| Skill                                  | KNTA_SKILL                  |
| Allow External Update of Actual Effort | KNTA_ALLOW_EXTERNAL_UPDATE  |
| _Scheduled Start Date                  | KNTA_SCHED_START_DATE       |
| _Actual Start Date                     | KNTA_ACTUAL_START_DATE      |
| _Scheduled Finish Date                 | KNTA_SCHED_FINISH_DATE      |
| _Actual Finish Date                    | KNTA_ACTUAL_FINISH_DATE     |
| _Scheduled Effort Over Duration        | KNTA_SCHED_EFF_OVER_DUR     |

# Index

## A

Additional Resources  
 Kintana documentation 6  
 Kintana education 9  
 Kintana services 9  
 Kintana support 10  
 Advanced Configuration  
 Guides 6  
 App Server Properties 96

## B

Budgets 97

## C

Command Conditions 18, 26  
 examples 18  
 Command Language 18  
 Command Steps 17  
 Commands  
 overview 11  
 triggering from Workflow  
 15  
 using 11  
 using with the workflow  
 15  
 Contact 97

## D

Documentation 6

## E

Entity Token  
 Accelerators 101  
 app server properties 96  
 budgets 97  
 command execution 102  
 contacts 97  
 Demand Management  
 Fields 127  
 distributions 98  
 Environment Applications  
 101  
 Environments 99  
 Notifications 103  
 organization units 104  
 Package Lines 106  
 Package Pending 107  
 PMO field group 127  
 Program 108  
 program reference field  
 group 128  
 Project 108  
 Project details 110  
 Project Field Group 127  
 Releases 110  
 Report Submissions 114  
 Request Details 113  
 Requests 111  
 Requests Pending 113  
 resource pools 115  
 Security Groups 116  
 Skills 116  
 staffing profile 117  
 Tasks 118  
 Tasks Pending 119  
 Users 121

work item field group 128  
 Workflow Steps 124  
 Workflows 123

Entity Tokens  
 Validations 122

## F

Field Group Tokens 126

## K

ksc\_begin\_script 80  
 example 81  
 ksc\_capture\_output 85  
 ksc\_clear\_exit\_value 88  
 ksc\_comment 79  
 ksc\_conc\_sub 79  
 example 79  
 ksc\_connect 62  
 ksc\_connect\_dest\_client 63  
 example 63  
 ksc\_connect\_dest\_server 63  
 example 64  
 ksc\_connect\_source\_client 64  
 example 65  
 ksc\_connect\_source\_server  
 65  
 examples 66  
 ksc\_copy 66  
 ksc\_copy\_client\_client 67  
 example 67, 68  
 ksc\_copy\_client\_server 68

example 68  
 ksc\_copy\_client\_tmp 71  
 ksc\_copy\_script 82  
 ksc\_copy\_script\_dest\_client 82  
 ksc\_copy\_script\_dest\_server 82  
 ksc\_copy\_script\_source\_client 83  
 ksc\_copy\_script\_source\_server 83  
 ksc\_copy\_server\_client 69  
   example 69  
 ksc\_copy\_server\_server 70  
   example 70  
 ksc\_copy\_server\_tmp 71  
 ksc\_copy\_tmp\_client 72  
 ksc\_copy\_tmp\_server 72  
 ksc\_end\_script 80  
   example 81  
 ksc\_exit 66  
 ksc\_gl\_migrate 86  
   example 86  
 ksc\_local\_exec 75  
 ksc\_om\_migrate 84  
   example 85  
 ksc\_parse\_jcl 87  
 ksc\_replace 76  
 ksc\_respond 73  
 ksc\_set 76, 87  
   example 75, 76, 77  
 ksc\_set\_env 77  
 ksc\_set\_exit\_value 87  
 ksc\_simple\_respond 73  
   examples 74  
 ksc\_store 76, 78, 87  
   example 78  
 ksc\_submit\_job 87

## O

Object Types  
   commands and Workflow 15  
 Ownership  
   setting for special commands 36

## R

Request Field Tokens 52  
   prefixes 52  
   table components 53

## S

Special Command  
   Parameters tab 24  
 Special Command Builder 29  
   using to build steps 38  
 Special Commands  
   adding parameters 25, 33  
   building steps with command builder 38  
   Commands tab 25  
   creating new 31  
   Deleting Parameters 35  
   editing parameters 35  
   header fields 23  
   ksc\_begin\_script 80  
   ksc\_capture\_output 85  
   ksc\_clear\_exit\_value 88  
   ksc\_comment 79  
   ksc\_conc\_sub 79  
   ksc\_connect 62  
   ksc\_copy 66  
   ksc\_copy\_script 82  
   ksc\_end\_script 80  
   ksc\_exit 66

ksc\_gl\_migrate 86  
 ksc\_local\_exec 75  
 ksc\_om\_migrate 84  
 ksc\_parse\_jcl 87  
 ksc\_respond 73  
 ksc\_run\_sql 88  
 ksc\_set 76, 87  
 ksc\_set\_env 77  
 ksc\_set\_exit\_value 87  
 ksc\_simple\_respond 73  
 ksc\_store 76, 78, 87  
 ksc\_submit\_job 87  
 nesting 39  
 ownership tab 29  
 parameters 89  
 predefined in Kintana 61  
 setting ownership 36  
 used by tab 30  
 user interface 22  
 using 37  
 window 23  
 Workbench 22

## T

Table Components  
   using tokens in 53  
 Token Builder Window 43  
 Token Evaluation 58  
 Tokens  
   building 48  
   default format 47  
   environment tokens 56  
   explicit entity format 47  
   field groups 126  
   formats 45  
   overview 41  
   parameter format 51  
   request fields 52  
   sub-entity format 55  
   user data format 50

within tokens 49

