



# **Kuali Coeus (KC) Installation Guide**

---

**Release v. 5.1**

**4/5/2013**

# Contents

Copyright and Licensing .....	i
Trademarks .....	i
Intellectual Property Management .....	i
Software Licensing .....	i
Contributor Licensing .....	i
Documentation Licensing .....	ii
Downloading, Installing, and Configuring .....	5
Installation Basics and High-Level Overview .....	5
Technical Requirements .....	6
Distribution Download .....	7
Quickstart Setup .....	8
Embedded Mode .....	12
Where to Find More KC Technical Documentation .....	14
Appendix A: Configuration Parameters .....	15

---

## Copyright and Licensing

All Kuali Foundation Licensing information is located online at: <https://wiki.kuali.org/display/KULFOUND/Licensing>

### Trademarks

Kuali® is a registered trademark™ of the Trustees of Indiana University. Other company and product names from contributor organizations may be trademarks of the respective companies with which they are associated.

### Intellectual Property Management

The Kuali Foundation has established a set of intellectual property management practices to protect the foundation, its members, and the extended Kuali community.

The Kuali Foundation uses various licenses to distribute software and documentation, to accept regular contributions from individuals and organizations, and to accept large grants of existing software products.

The sections that follow explain Kuali copyright information as it pertains to the following three licensing areas:

- Software Licensing
- Contributor Licensing
- Documentation Licensing

**Intellectual Property Contact Information:** If you have any questions about Kuali Intellectual property, please contact The Kuali Foundation at [licensing@kuali.org](mailto:licensing@kuali.org).

### Software Licensing

#### Acknowledgments



---

Copyright © 2007-2013 The Kuali Foundation. All rights reserved. Kuali Coeus is licensed for use pursuant to the Educational Community License, Version 2.0 (<http://www.opensource.org/licenses/ec12.php>). Portions of Kuali Coeus copyrighted by other parties, including the parties listed below, and you should see the licenses directory for complete copyright and licensing information. Questions about licensing should be directed to [license@kuali.org](mailto:license@kuali.org).

---

### Contributor Licensing

#### Acknowledgments

Copyright 2007-2013 The Kuali Foundation. All rights reserved. Kuali Coeus is licensed for use pursuant to the Educational Community License, Version 2.0. Portions of Kuali Coeus copyrighted by other parties, including the parties listed below, and you should see the licenses directory for complete copyright and licensing information. Questions about licensing should be directed to [license@kuali.org](mailto:license@kuali.org).

#### Third Party Contributions

Portions of Kuali Coeus were developed by Indiana University, Cornell University, Michigan State University, University of Arizona, Massachusetts Institute of Technology, Colorado State University, Iowa State University, University of California - Berkeley, University of California - Davis, West Virginia University, Coeus Consortium, and Huron Consulting. (<http://www.kuali.org/kc>).



This product includes software developed by the Apache Software Foundation (<http://www.apache.org>).

This product includes software developed by ANTLR (<http://www.antlr.org/>).

This product includes software developed by the JAX-RPC Project part of Project GlassFish (<https://jax-rpc.dev.java.net/>).

This product includes software developed by the SAAJ Project part of Project GlassFish (<https://saaj.dev.java.net/>).

This product includes software developed by Displaytag (<http://displaytag.sourceforge.net/11/>).

This product includes software developed by the JDOM Project (<http://www.jdom.org/>).

This product includes software developed by the University Corporation for Advanced Internet Development Internet2 Project (<http://www.ucaid.edu>).

This product includes software developed by the Open Symphony Group (<http://www.opensymphony.com/>).

This product includes software developed by the Indiana University Extreme! Lab (<http://www.extreme.indiana.edu/>).

This product includes software developed by the SAXPath Project (<http://www.saxpath.org/>).

This product includes software developed by the JA-SIG Collaborative (<http://www.ja-sig.org/>).

This product includes icons created by Mark James. (<http://www.famfamfam.com/lab/icons/silk>)

This product includes software licensed under GNU Lesser General Public License (<http://www.opensource.org/licenses/lgpl-license.php>).

This product includes software licensed under Common Development and Distribution License (<http://www.opensource.org/licenses/cddl1.php>).

This product includes software licensed under Common Public License (<http://www.opensource.org/licenses/cpl1.0.php>).

This product includes software licensed under the Mozilla Public License (<http://www.mozilla.org/MPL/>).

This product includes the Kuali Rice module licensed under the Kuali Foundation ECL 2.0.

Portions Copyright (c) 2000-2006 The Legion of the Bouncy Castle. All Rights Reserved. (<http://www.bouncycastle.org>)

Portions Copyright (c) 2000-2005 INRIA, France Telecom. All Rights Reserved.

Portions Copyright (c) 2001-2006 Sun Microsystems, Inc. All Rights Reserved.

Portions Copyright (c) 2003-2006 The Werken Company. All Rights Reserved. (<http://jaxen.codehaus.org/>)

Portions Copyright (c) 2001-2005 MetaStuff, Ltd. All Rights Reserved. (<http://www.dom4j.org/>)

## Documentation Licensing



**Copyright © 2013 by The Kuali Foundation. Some rights reserved.**

Kuali Coeus User Documentation by the [Kuali Foundation](#) is licensed under a [Creative Commons Attribution-Share Alike 3.0 United States License](#). Permissions beyond the scope of this license may be available at <http://www.kuali.org>.

## Creative Commons License Deed



**Attribution-Share Alike 3.0 United States (CC BY-SA 3.0)**

**You are free:**



**to Share** – to copy, distribute, and display the work



**to Remix** – to make derivative works

### Under the following conditions:



**Attribution.** You must attribute this work to the [Kuali Foundation](http://www.kuali.org) (licensor), but not in any way that suggest that they endorse you or your use of the work.



**Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.



**Permissions beyond** the scope of this public license are available at <http://www.kuali.org>.

- **Notice** - For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is by including the information in this section of this document, and including a link to <http://www.kuali.org>.
- Any of the above conditions can be waived if you get permission from the [Kuali Foundation](http://www.kuali.org), the copyright holder.
- Apart from the remix rights granted under this license, nothing in this license impairs or restricts the author's moral rights. The Kuali Foundation (licensor) retain the right to request removal of their name from a derivative or collective work they don't like – for example, when the author believes a derivative work represents a “derogatory treatment” of the author's work.

### Disclaimer

**Your fair use and other rights are in no way affected by the above.**

The Commons Deed is not a license. It is simply a summary and handy reference for understanding the [Legal Code](#) (the full license) — it is a human-readable expression of some of its key terms. Think of it as the user-friendly interface to the Legal Code beneath. This Deed itself has no legal value, and its contents do not appear in the actual license. Creative Commons is not a law firm and does not provide legal services. Distributing of, displaying of, or linking to this Commons Deed does not create an attorney-client relationship.

The information in this document is subject to change without notice. It has been obtained from sources believed to be reliable. Although the author and the Kuali Foundation have made every effort to ensure the accuracy of this document, neither the authors nor the publisher assumes any liability or responsibility for any inaccuracy or omissions contained therein, or for any loss or damage arising from the information presented.

If you discover any issues with the documentation, please report your findings, in writing, to The Kuali Foundation. The Kuali Foundation does not warrant that this document is error-free.

Bradley C. Wheeler, Franklin Hall 116, 601 East Kirkwood Avenue, Indiana University, Bloomington, Indiana 47405

All other products or company names represented in this document are not to be viewed as endorsements by The Kuali Foundation (the licensor), and may be trademarks for their respective owners.



---

# Downloading, Installing, and Configuring

## Installation Basics and High-Level Overview

This topic describes how to install Kuali Coeus on a single server. The Kuali Coeus application binaries and source code are included as part of the installation distribution.

The Kuali Coeus application assists in the accessibility, administration and support of research information. There are many components that are working together to configure and install Kuali Coeus. The core system is known as Kuali Rice (KR), which is a set of integrated middleware products that allows Kuali Coeus to interface with technical modules running from a web server. Kuali Rice facilitates enterprise workflow functionality and provides customizable and configurable user interfaces that have a clean and universal look and feel. Kuali Rice contains five core modules of which two are actively used in Kuali Coeus. Kuali Enterprise Workflow (KEW) monitors document routing and approval while automating electronic processes and transactions across the enterprise. The Kuali Nervous System (KNS) alternately handles functionality common to many modules and can be thought of as a framework of reusable components providing an abstraction layer for developers to easily integrate with other Kuali Rice components.

Kuali Coeus scales depending upon your environment needs and therefore allows for various architectural choices to be implemented during installation. This topic covers the essential architectural components that are vital to the overall operation of the system. Kuali Coeus handles data storage using a database to maintain records. Care must be taken to ensure that the database remains accessible during times when Kuali Coeus is active. The Kuali Coeus application server manages data for presentation through a web browser.



## Technical Requirements

Kuali Coeus is versatile in its ability to deploy on numerous platforms but has only been tested on a select set of platforms. This section describes specific requirements to install Kuali Coeus.

### System Server

This section outlines the minimum **server requirements** for Kuali Coeus.

- **Hardware:** Recommended minimum hardware for the web application is 2 cores of a recent CPU architecture along with 3 gig of reserved RAM. Minimum hardware and settings for the database should be based on vendor recommendations, but there should be high bandwidth with low latency between the web application and database as this is a database intensive application.
- **Operating Systems:** Any OS which supports either MySQL or Oracle databases should work, including Solaris, but it has been run on Windows, Linux (Ubuntu and RedHat distributions), and MacOSX.
- **Web Application Platform:** The Web application platform can be any fully compliant modern J2EE container. Testing has been done with Jetty 7 and Tomcat 6.
- **Heap Space:** The Webapp generally requires 1 gig of heap space (-Xmx1g) and 256 meg of perm gen space (-XX:MaxPermSize=256m) to start and run. For production use, at least 2 gig of heap and 512 meg of perm gen space are recommended.
- **Other Recommended Settings:** Other recommended settings are -server (to improve resource utilization for long-running tasks) and -XX:+UseConcMarkSweepGC (to improve garbage collection performance).

### Supporting Application Software

This section outlines the minimum **software requirements** for Kuali Coeus.

- Oracle Java Development Kit (JDK 1.6.x)
- Web Application Server (Apache 2.x)
- Servlet container (Apache Tomcat 6.x)
- Web Browser (Firefox 1.5.x, Internet Explorer 6.x or Safari 2.0.4.x)
- Apache Ant 1.7.x
- Maven 3.0.x
- Oracle Database 10g Release 2 (10.2.0.x)
- Oracle Database 10g Release 2 (10.2.0.x) JDBC Driver
- MySQL 5.x
- MySQL Connector/J





## Database

This section outlines the minimum database **requirements** for Kuali Coeus.

KC has been tested against Oracle and MySQL. Supported versions of database products that Kuali Coeus has been tested with are as follows:

- Oracle10g/11g: JDBC drivers for Oracle can be obtained from <http://www.oracle.com/index.html>
- MySQL: JDBC drivers for MySQL can be obtained from [www.mysql.com](http://www.mysql.com)

KC requires a User Account with access **permissions** to the database to perform the following:

- Connect to the database
- Create, Alter and Drop Tables, Triggers, Views, Procedures and Sequences
- Insert, Update and Delete data in the database tables

KC recommends the following database **settings**:

- For MySQL running on a case-sensitive OS (i.e., Linux) it is recommended to use settings to force all table names to lowercase (see other MySQL recommended settings in the Rice documentation). MySQL database should have default UTF8 encoding.

## Distribution Download

To download Kuali Coeus, navigate to the following site and fill out the download form appropriately:

<http://www.kuali.org/download-form>

The Kuali Coeus distribution is available from the Kuali Foundation at

[https://svn.kuali.org/repos/kc\\_project/tags/kc-project-5\\_0-tag/](https://svn.kuali.org/repos/kc_project/tags/kc-project-5_0-tag/)

The Kuali Coeus download contains:

- Kuali Coeus source code
- Database scripts – Full install, 3.0 to 5.0 upgrade, 3.0.1 to 5.0 upgrade, 3.1 to 5.0 upgrade, 3.1.1 to 5.0 upgrade, 3.2 to 5.0 upgrade, and 4.0 to 5.0 upgrade
- Binary distribution
- Quickstart Startup Guide
- Project licenses
- Integrated Application Help (HTML)
- Release Notes
- List of known bugs

Extract the ZIP file that you downloaded to the folder of your choice.

## Quickstart Setup

Use the following installation instructions to guide you through the baseline out-of-the-box setup.



---

Information on more **advanced configuration and customization** can be found in the [public documentation](#).

---



**Before You Begin** - The following are required:

- Java 1.6
- Servlet 2.5 / JSP 2.1 Compatible Servlet Container such as Apache Tomcat 6
- Oracle database instance or MySQL database instance



**To set up KC (and Rice) database:**

KC database release contains all SQL scripts needed to install a new schema or upgrade a KC 3.1.1, 5.0 or 5.0.1 database using Rice bundled or embedded mode with the database objects (tables, constraints, bootstrap data) for the KC application. If you are on a different version of KC then you will need to use previous release packages to get to one of the levels supported by this release.

If installing in an embedded Rice environment you will be prompted to install/upgrade Rice. If your Rice server has been upgraded already you will answer no to the question (KC-specific data will still be added to your Rice server).

### Installation Steps for Oracle:



---

Installation has been tested with sqlplus. Other tools may require modification of the scripts.

---

1. Create Oracle username of less than 8 characters
2. Make sure Oracle user has following privileges:
  - DEFAULT TABLESPACE <desired tablespace (typically Users)>
  - QUOTA UNLIMITED ON <desired tablespace (typically Users)>
  - CREATE SESSION
  - CREATE SYNONYM
  - CREATE PROCEDURE
  - CREATE TRIGGER
  - CREATE TABLE
  - CREATE TYPE
  - CREATE VIEW
  - CREATE SEQUENCE



---

A user's DEFAULT TABLESPACE is set with the CREATE USER statement or ALTER USER statement. The TABLESPACE should not be the SYSTEM tablespace.




---

A New install will COMPLETELY clear data from any existing KC tables in this schema!

---

### Installation Steps for MySQL:


---

 Installation has been tested with the mysql client. Other tools may require modification of the scripts.

---

1. Set the following settings in MySQL:
  - max\_allowed\_packet=1M
  - transaction-isolation=READ-COMMITTED
  - lower\_case\_table\_names=1
  - max\_connections=1000
2. Create MySQL username of less than 8 characters.
3. Create MySQL default schema for username with the default character set of UTF8. If a different character set is desired, the ddl scripts will need to be updated to the new character set.
4. Make sure MySQL user has following privileges on the schema
  - Select
  - Insert
  - Update
  - Delete
  - Create
  - Drop
  - Index
  - Alter
  - Create\_view
  - Create\_routine
  - Alter\_routine
  - Create\_tmp\_table
  - Lock\_tables

---

 Be sure to start with an empty schema before doing a NEW install.

---

### To populate data into KC (and Rice) database:

#### Installation Steps for Both Oracle and MySQL

1. Run the appropriate install script in db\_scripts/main (KC\_Install.bat for Windows or KC\_Install.sh for Linux/Unix).
2. You will be prompted for the following:
  - Rice Mode = Choose one: Bundle, Embed
  - DB\_Server = Choose one: Oracle, MySQL
  - Version = Choose one: New, 3.1.1, 5.0, 5.0.1
  - username = The KC database username name to install database scripts with (bundled Rice goes here too).
  - password = the password for username
  - DB\_server\_name = the tns name (Oracle) or schema name (MySQL) used to connect to KC database.

If embedded mode selected the following will be asked:

- Rice Username = the Rice database username to install rice scripts with (embedded Rice).
- Rice Password = password for Rice username
- Rice DB Server name = the tns/schema name used to connect to Rice database.

- Do you wish to Install/Upgrade your Embedded Rice Server (Don't perform if your server was upgraded separately)
- 3. Review .log files for installation errors (in LOGS directory).



#### To set up KC Application:

1. Be sure database is setup and populated with KC and Rice data (as instructed above).
2. Place binary/kc-config.xml in one of the configured default locations: {Userhome}/kuali/main/ptd/kc-config.xml OR /opt/sa\_forms/java/ptd/kra/kc-config.xml.



If you are upgrading and have an existing kc-config.xml in this location, please replace it and merge any customizations, as new parameters are often added.

- On Windows with Tomcat running as a service, the best location is usually c:\kuali\main\ptd
- To use a different location, open kc-ptd.war and create an "alt.config.location" parameter in /WEB-INF/classes/META-INF/kc-config-build.xml.
- You SHOULD also be able to pass a java parameter of alt.config.location to your j2ee container (not guaranteed to work)



This file also contains the "build.environment" parameter referenced in the external kc-config.xml file as "environment"

3. Edit parameters in the external kc-config.xml (that you just placed in step #2) as desired.
  - You will at least need to fill in the following parameters with your database connection information: datasource.url, datasource.username, datasource.password.
  - You will also need to change application.host to your server's host name (i.e. kuali.yourinstitution.edu).
  - Other commonly changed parameters are: application.http.scheme, http.port and app.context.name (be sure app.context.name matches your deployed webapp -- normally the name of your war file)
  - If you are using SSL, be sure that application.http.scheme and http.port reflect this.
  - If you are using MySQL, see the MySQL changes at the end of this guide.

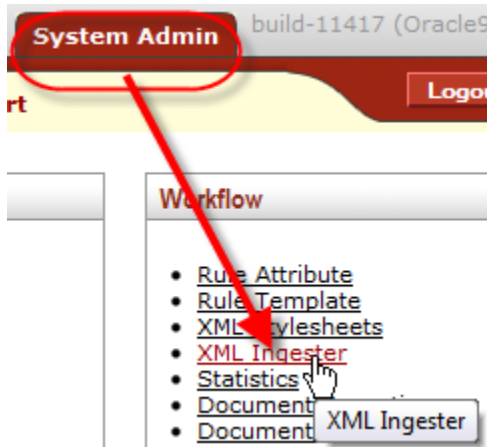


For embedded mode or further configuration, please consult the **Embedded Mode** section.

4. Obtain the database driver.
  - **For Oracle:** Obtain ojdbc14.jar from an oracle client install ({OracleHome}/jdbc/lib) or the Oracle website (<http://www.oracle.com>).
  - **For MySQL:** Obtain the jar file for the appropriate MySQL jdbc driver from mysql.com.
5. Copy the jdbc jar file into the common jar file library.
  - Example: For Tomcat 6, copy ojdbc6.jar into {tomcat install dir}/lib.
6. Deploy kc-ptd.war to your servlet container (see user guide for your particular servlet container).
7. The KC application should now be accessible at: {application.host}/{app.context.name} (as specified in the external kc-config.xml -- app.context.name is kc-ptd, if not overridden).
8. Login as 'admin' (click on a tab or option and the login screen will appear).

9. Ingest the KEW documents.

- Click the System Admin tab and then select XML Ingestor from the Workflow pane.



- Find the KEW zip files in the distribution package (they are located in subdirectories under db\_scripts)
- Install Rice-KEW.zip followed by Full-KC-KEW.zip.
- If you have any locally modified KEW files, you should compare with the contents of these zip files and merges changes, as necessary. If your local KEW files have replaced any existing files and require no updates then simply remove the from the zip so that they are not overwritten.
- MySQL changes to external kc-config.xml file -- adjust datasource.url as needed to fit your install:

```
<param name="datasource.url">jdbc:mysql://localhost:3306/kcdev</param>
<param name="datasource.objb.platform">MySQL</param>
```

10. **(Optional)** Install demonstration data. Convenient install scripts for demonstration data can be found in the db\_scripts/main folder (KC\_Install\_Demo.sh/bat). The demonstration data is only guaranteed to install cleanly into a new KC install with the server halted. If you have previous test data or demonstration data then you will encounter errors, which may or may not affect functionality.



Demonstration data is for testing functionality of the software only and should not be included in an actual implementation. Take a backup of your database before applying demonstration data or be prepared to reinstall from scratch before implementation.

## Configuration Parameters



**Tip:** Refer to **Appendix A: Configuration Parameters** within this documentation set for supplementary information that will prove helpful by providing default values and handy tips for both KC-specific and Rice parameters.

## Embedded Mode

Embedded mode allows you to run KC with a central Rice server. This has the benefit of a central Doc Search, Action List, and KIM maintenance.

It also allows some processing to be offloaded to the rice server, which provides different scaling characteristics. Embedded mode is currently the preferred deployment model.



---

For a more detailed description of embedded mode, see the Rice documentation [<https://wiki.kuali.org/display/KULRICE/Documentation>].

---

## Standalone Rice

1. Create/Configure Rice Server Database
2. Configure Standalone Rice Server (see example-config/rice-config.xml in the rice project)
3. Startup Rice server

## Running KC Client Application In Embedded Mode

**Before you begin:**

1. Create/Configure KC Database (This will include the Rice Client tables and KC's tables).
2. Setup KC database in Embedded Mode.
  - Run the Database setup script
  - Select EMBED for your Rice mode
  - For Install/Upgrade Embedded Rice Server Side
    - Select Y if you need to cleanly install a new Rice database along with KC data
    - Select N if you need to upgrade an existing Rice database with KC data
3. Configure KC.

**To run KC in Embedded Mode:**

- Point KC to the client & server databases
- Configure the Rice URL
- Turn off dev-mode
- Set the run modes to embedded for the Rice modules
- Configure a valid keystore for secure communication with Rice
- If you are upgrading and you override your encryption.key parameter, you may need to update it in order to access older workflow documents.
  - Download the tool from <http://svn.kuali.org/repos/rice/tools/encryption-key-converter/trunk>
  - run mvn package from the command line to create the executable jar in the target folder
  - Run the executable in the target directory with "java -jar encryption-key-converter-1.0-SNAPSHOT.jar <current encryption.key property value>" to output the new key value to put in your rice-config.xml

## Embedded Mode Configuration File

```
<!-- KC Client DB -->
<param name="datasource.url">jdbc:oracle:thin:@127.0.0.1:1521:XE</param>
<param name="datasource.username">KCDEV</param>
<param name="datasource.password">secret</param>
<param name="datasource.ojb.platform">Oracle9i</param>

<!-- For Embedded Mode -->
<param name="rice.server.url">http://127.0.0.1:8090/kr-dev</param>

<param name="kim.mode">EMBEDDED</param>
<param name="kcb.mode">REMOTE</param>
<param name="kew.mode">EMBEDDED</param>
<param name="ken.mode">REMOTE</param>
<param name="ksb.mode">REMOTE</param>
<param name="kns.mode">LOCAL</param>
<param name="krad.mode">REMOTE</param>
<param name="krms.mode">REMOTE</param>
<param name="coreservice.mode">REMOTE</param>
<param name="location.mode">REMOTE</param>
<param name="core.mode">LOCAL</param>
<param name="dev.mode">>false</param>
<param name="rice.portal.allowed.regex">
^(${application.url}|${rice.server.url})(/.*)
</param>
<param name="context.names.rice">kr-dev</param>
<!--This displays the rice server config in the system admin tab-->
<param name="rice.portal.links.showRiceServerConfig">>true</param>

<!-- Rice Server DB -->
<param name="server.datasource.url">jdbc:oracle:thin:@127.0.0.1:1521:XE</param>
<param name="server.datasource.username">RICEDEV</param>
<param name="server.datasource.password">secret</param>
<param name="server.datasource.ojb.platform">Oracle9i</param>

<!-- Keystore Configuration -->
<param name="keystore.file">${user.home}/kuali/main/${environment}/rice.keystore</param>
<param name="keystore.alias">rice</param>
<param name="keystore.password">r1c3pw</param>
```

4. Startup KC.
5. Verify the configuration.

**To verify the configuration, try the following:**

- Create a proposal document. Save it. Close it. From the main portal page click on Action List -> Find the Proposal -> Log -> Future Action Requests
- Go to System Admin Tab. Try Person Maintenance or Parameter Maintenance. These pages should work correctly and they should link to the central rice server.



## Where to Find More KC Technical Documentation

The following is a list of links to other sources of information related to Kuali Coeus installation, and references to additional online resources where you can find more information that may prove to be helpful for installation activities:

### ***Kuali Resources***

- **KC Technical Documentation:** <https://wiki.kuali.org/display/KRACOEUS/KC+Technical+Documentation>
- **Kuali Rice Documentation:** <https://wiki.kuali.org/display/KULRICE/Documentation>
- **KC Implementation Collaboration Group:** <https://wiki.kuali.org/display/KRADOC/Collaboration>
- **KC System Requirements:** <https://wiki.kuali.org/display/KRADOC/System+Requirements>
- **Grants.gov Server Configuration:**  
<https://wiki.kuali.org/display/KRADOC/Grants.gov+Server+Configuration>
- **Kuali Coeus Implementation SIG:**  
<https://wiki.kuali.org/display/KRADOC/Kuali+Coeus+Implementation+SIG>
- **KC Technical Toolset:** <https://wiki.kuali.org/display/KRADOC/KC+Technical+Toolset>
- **Kuali Coeus Research Administration SIG wiki page:**  
<https://wiki.kuali.org/display/KRACOEUS/Collaboration>
- **Kuali Architecture and Development Standards:** <http://kuali.org/files/pdf/KualiStandards.pdf>
- **KC Test Drive:** <http://testdrive.kc.kuali.org/kc-ptd/portal.jsp>

### ***Tomcat Resources***

- **The Apache Tomcat 6.0 Servlet/JSP Container:** <http://tomcat.apache.org/tomcat-6.0-doc/index.html>
- **The Apache Jakarta Project:** <http://jakarta.apache.org/>

### ***Eclipse Resources***

- **Eclipse Documentation:** <http://www.eclipse.org/documentation/>

### ***Oracle Resources***

- **Oracle Database Express Edition Getting Started Guide:**  
[http://download.oracle.com/docs/cd/B25329\\_01/doc/admin.102/b25610/toc.htm](http://download.oracle.com/docs/cd/B25329_01/doc/admin.102/b25610/toc.htm)
- **Oracle Express Edition Tutorial:** <http://st-curriculum.oracle.com/tutorial/DBXETutorial/index.htm>



# Appendix A: Configuration Parameters

This topic provides information that pertains to the following **two types of configuration parameters**:

- **KC-Specific** - Parameters defined specifically for KC
- **KC-Customized Rice** - Customizations of parameters provided by Rice

The following information is provided for **each parameter**:

- **Name** – The parameter name, for example, ‘build.properties’
- **Purpose** – A textual description of the parameter’s purpose
- **Default Value** – The pre-configured value prior to customization
- **Notes** – Information that aids in the understanding of proper usage

## ***build.environment***

**Purpose:** The environment the application is running in (ex: dev, prd, tst).

**Default Value:** dev

**Notes:** Since the Rice param environment=\${build.environment}, this parameter is indirectly used by rice to construct urls and resolve config files.

## ***build.version***

**Purpose:** The version number displayed on the application header.

**Default Value:** NO VERSION

**Notes:** Since the rice param version=\${build.version}, this parameter is indirectly used by rice for display purposes.

## ***kim.mode***

**Purpose:** Used to set the runmode of the KIM module.

**Default Value:** local

**Notes:** KC currently supports local (bundled) & embedded. We have not tested in a mixed runmode configuration. See the rice documentation for mode information on the various runmodes (deployment models).

### ***kcb.mode***

**Purpose:** Used to set the runmode of the kcb module.

**Default Value:** local

**Notes:** (see kim.mode)

### ***kew.mode***

**Purpose:** Used to set the runmode of the kew module.

**Default Value:** local

**Notes:** (see kim.mode)

### ***ken.mode***

**Purpose:** Used to set the runmode of the ken module.

**Default Value:** local

**Notes:** (see kim.mode)

### ***dev.mode***

**Purpose:** Used when setting the runmode.

**Default Value:** TRUE

**Notes:** set to false when running in embedded mode. See rice documentation for more details.

### ***rice.server.url***

**Purpose:** The url to the rice server.

**Default Value:** \${application.url}

**Notes:** For local (bundled mode) this defaults to the kc client url. For embedded mode this will point to a standalone server.

### ***context.names.rice***

**Purpose:** The rice server context.

**Default Value:** kc context

**Notes:**

### ***kra.externalizable.images.url***

**Purpose:** The url to kc specific images.

**Default Value:** /\${app.context.name}/static/images/

**Notes:**

### ***core.rice.struts.message.resources***

**Purpose:** rice's message bundle(s)

**Default Value:** KR-

ApplicationResources,org.kuali.rice.kew.ApplicationResources,org.kuali.rice.ksb.messaging.Applicat  
ionResources,KIM-ApplicationResources

**Notes:** (see rice.struts.message.resources)

### ***kc.struts.message.resources***

**Purpose:** KC's message bundle(s)

**Default Value:** ApplicationResources

**Notes:** (see rice.struts.message.resources)

### ***rice.struts.message.resources***

**Purpose:** Message bundle names to load.

**Default Value:** \${core.rice.struts.message.resources}, \${kc.struts.message.resources}

**Notes:** In the rice parameter rice.struts.message.resources messages in the bundles on the right will override the same messages in bundles on the left. Given the default values, if a property [display.name=the display name] exists in the org.kuali.rice.ksb.messaging.ApplicationResources and ApplicationResources (kc's message bundle) kc's message will take precedent.

### ***datasource.url***

**Purpose:** the url of the client (or client & rice) database

**Default Value:** jdbc:oracle:thin:@127.0.0.1:1521:KUALI

**Notes:** in embedded mode this will be the KC database only. In bundled mode this will be the KC database & the rice database. See rice documentation for additional details

### ***datasource.username***

**Purpose:** username of the client (or client & rice) database

**Default Value:** KRADEV

**Notes:** (see datasource.url)

### ***datasource.password***

**Purpose:** password of the client (or client & rice) database

**Default Value:** ask your team

**Notes:** (see datasource.url)

### ***datasource.platform***

**Purpose:** platform of the client (or client & rice) database

**Default Value:** Oracle9i

**Notes:** (see datasource.url)

### ***server.datasource.url***

**Purpose:** rice server database property

**Default Value:** \${datasource.url}

**Notes:** in embedded mode this will be the Rice database only. In bundled mode this will be the KC database & the rice database. See rice documentation for additional details

### ***server.datasource.username***

**Purpose:** rice server database property

**Default Value:** \${datasource.username}

**Notes:** (see server.datasource.url)

### ***server.datasource.password***

**Purpose:** rice server database property

**Default Value:** \${datasource.password}

**Notes:** (see server.datasource.url)

### ***server.datasource.ojb.platform***

**Purpose:** rice server database property

**Default Value:** \${datasource.ojb.platform}

**Notes:** (see server.datasource.url)

### ***server.datasource.platform***

**Purpose:** rice server database property

**Default Value:** \${datasource.platform}

**Notes:** (see server.datasource.url)

### ***server.datasource.driver.name***

**Purpose:** rice server database property

**Default Value:** \${datasource.driver.name}

**Notes:** (see server.datasource.url)

### ***server.datasource.pool.validationQuery***

**Purpose:** rice server database property

**Default Value:** \${datasource.pool.validationQuery}

**Notes:** (see server.datasource.url)

### ***server.datasource.pool.maxWait***

**Purpose:** rice server database property

**Default Value:** \${datasource.pool.maxWait}

**Notes:** (see server.datasource.url)

### ***server.datasource.pool.minSize***

**Purpose:** rice server database property

**Default Value:** \${datasource.pool.minSize}

**Notes:** (see server.datasource.url)

### ***server.datasource.pool.maxSize***

**Purpose:** rice server database property

**Default Value:** \${datasource.pool.maxSize}

**Notes:** (see server.datasource.url)

### ***server.datasource.pool.size***

**Purpose:** rice server database property

**Default Value:** \${datasource.pool.size}

**Notes:** (see server.datasource.url)

### ***server.datasource.connectionTimeout***

**Purpose:** rice server database property

**Default Value:** \${datasource.connectionTimeout}

**Notes:** (see server.datasource.url)

### ***keystore.alias***

**Purpose:** The alias for the rice keystore file.

**Default Value:** onestartshareservices-devandtst

**Notes:** required for kc client application to communicate with a standalone rice server (embedded mode)

### ***keystore.file***

**Purpose:** The path & file name to the rice keystore file.

**Default Value:** kul

**Notes:** (see keystore.alias)

### ***keystore.password***

**Purpose:** The password for the rice keystore file.

**Default Value:** kulpass

**Notes:** (see keystore.alias)

### ***filter.login.class***

**Purpose:** specifies a servlet filter class named login

**Default Value:** org.kuali.rice.kew.web.DummyLoginFilter

**Notes:** This class will probably change to a single sign on solution. This class should not be used in production.

### ***filtermapping.login.1***

**Purpose:** maps all requests to go through the login filter

**Default Value:** /\*

**Notes:**

### ***application.custom.image.url.copyoneper***

**Purpose:** Specifies a non-default location for an image in question framework

**Default Value:** /\${app.context.name}/static/images/buttonsmall\_copyallperiods.gif

**Notes:** this probably doesn't need to change

### ***application.custom.image.url.copyallperiods***

**Purpose:** Specifies a non-default location for an image in question framework

**Default Value:** /\${app.context.name}/static/images/buttonsmall\_copyoneper.gif

**Notes:** this probably doesn't need to change

### ***s2s.keystore.location***

**Purpose:** The absolute path to Keystore file for grants.gov.

**Default Value:** path to keystore

**Notes:** The certificate imported to this keystore is being used to connect and getting authenticated to Grants.Gov server. The certificate should be CSA certified and should be registered with Grants.Gov before using it.

### ***s2s.keystore.password***

**Purpose:** Password for the keystore

**Default Value:** keystore password

**Notes:**

### ***s2s.truststore.location***

**Purpose:** Absolute path to the truststore file, where all public server certificates which application has access to.

**Default Value:** path to truststore

**Notes:** To access grants.gov server, the truststore file should have grants.gov web service server's public certificate imported in it.

### ***s2s.truststore.password***

**Purpose:** Password for the truststore file.

**Default Value:** truststore password

**Notes:**

### ***grants.gov.s2s.host***

**Purpose:** Host url to access s2s web service

**Default Value:** grants.gov host



**Notes:** To access test server use <https://atws.grants.gov:446/app-s2s-server/services>  
To access production server use <https://ws.grants.gov:446/app-s2s-server/services>  
Default value is set to point to Grants.Gov TEST server

### ***grants.gov.s2s.port***

**Purpose:** S2S web service port, where all major services are declared.

**Default Value:** grants.gov port

**Notes:** Since this is a variable defined in Grants.Gov reference implementation, we also keep it as a variable. Usually, we do not change this property.