

Kuwaiba Open Inventory user's Manual

Neotropic SAS

27.07.2016

System Version **1.0**

Visit **kuwaiba.org** for documentation, latest updates and upcoming events

Contents

Document History

License

Introduction 1

Connection to the Server 2

Data Model Manager 4

List Type Manager 5

Document History

Date	Comments
September 28th 2010	First issue shipped with version 0.1.1
November 26th 2010	Update to cover the new features in 0.2
December 26th 2010	Update to cover the new features in 0.2.1
January 18 th 2001	Changes in version 0.3 alpha
February 3 rd 2011	Changes in version 0.3 beta
March 13 th 2011	Changes in version 0.3 beta 2
May 16th 2011	Changes in version 0.3 stable (the clear button in the graphical query editor)
May 23rd 2012	Adapted to version 0.4
October 23rd 2012	Adapted to version 0.5
June 4 th 2013	Added documentation for Pools module
June 12 th 2013	Added documentation for Data model Manager module and some other minor changes
January 19 th 2015	Adapted manual for version 0.7
November 6 th 2015	Added documentation about bulk upload, software asset management and detailed physical connections
July 27th 2016	Adapted to Kuwaiba version 1.0. LaTeX is now used instead of LibreOffice to create the documentation.

License



This document is published under the terms of a license Creative Commons by-nc-sa. You can find details about it at <http://creativecommons.org/licenses/by-nc-sa/2.0/>



Kuwaiba Server and Client are licensed under EPL v1 and GPL v2. You can find the whole text of this licenses at <http://www.eclipse.org/legal/epl-v10.html>
<http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>

Disclaimer

- Netbeans and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. The Kuwaiba project is not endorsed to any of them.
- This document is provided “as is”, with no warranty at all. Install the software and follow the instructions included at your own risk.
- Kuwaiba uses third-party components with compatible open source licenses (LGPL, BSD-like, etc). You can find a complete list at the project’s web page.

Introduction

Kuwaiba sees an inventory system as a living entity, not growing only in terms of size, but also in structure and intelligence. The main reason is that business requirements change constantly and therefore, the application must be ready to respond to new scenarios. One of the key concepts that can help you unlock the potential of Kuwaiba is the **data model**. It provides a simplified representation of the network and the business from an operational point of view. It can be seen as the skeleton that supports the application, but a skeleton from which you can add, remove and change elements as you go. Later in this document you will be able to see what tools you can use to manage it. For now, just keep in mind that the better you design your data model and the more you get to know it, the more you will take advantage of the application.

Having said that, you will find four types of resources in a typical data model:

- **Physical:** Equipment, pipes, cables, fiber optics, facilities, parts and in general every physical asset from a port to a building.
- **Logical:** These are all the resources related to non-tangible technology assets. In this group fits timeslots, virtual circuits, VLANs, disk space, available bandwidth, etc.
- **Other Non-physical:** mostly software-related assets, such as licenses or virtual machines.
- **Administrative:** These are all those related to administrative tasks, human resources or commercial management. Customers, their services, SLAs (and related parameters like availability or throughput), sales and technical staff assigned to those services, vendors and states belong to this category.

The Kuwaiba desktop client is a set of views (trees, topologies, editors) that allow to put together these elements based on business rules and user-defined models. Kuwaiba extends the concept of **CMDDB** (Configuration Management Database, a place where you store objects that can hold configuration information or be subject to configuration themselves -so called Configuration Items- and their relationships) and enables you to perform network design tasks, support capacity management and provisioning workflows and assist field and customer service teams to improve response times.

Kuwaiba helps you model your network according to your needs, no matter if you're an ISP, a carrier or just a guy with a large (or small!) IT infrastructure to manage. It's open source, under active development and new models are added every release. You can contribute to the project by providing technical insight on a particular technology, testing, translating or just sending your feedback through forums¹ and mailing lists².

¹Forums <https://sourceforge.net/p/kuwaiba/discussion/>

²mailing lists <https://sourceforge.net/p/kuwaiba/mailman/>

Connection to the Server

The first thing you will see when opening the client is the window in the figure 1. The default user and password are **admin/kuwaiba**.

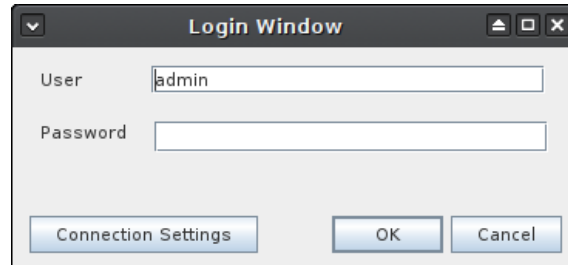


Figure 1: Authentication window

The default connection settings should be enough if the server is running on the same computer the client is. If that's not the case, open the Connection Settings window (figure 2).

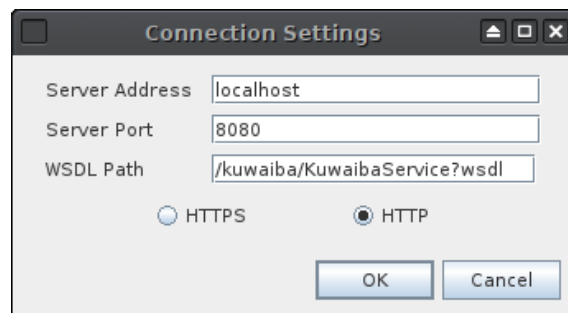


Figure 2: Connection Settings window

- **Server Address** Refers to the server IP address or canonical name.
- **Server Port** is the port Glassfish (the application server) is listening to.
- **WSDL Path** is the path within the application server the web service interface definition can be found. Usually this value should remain unchanged.
- Protocol is the transport protocol to be used. By default is HTTP, but is highly advisable to request your administrator to setup a secure connection, otherwise your credentials will be transmitted in plain text over the network.

Except for the password, the last successful settings will be saved upon clicking OK.

Important

If you are unsure if the server is reachable from your location, open a browser and type the address: **http://[server_address]:[server_port]/[wsdl_location]**

You should see a large XML document.

Troubleshooting

- For a **Can't contact backend** error, check the Administrator's Manual Troubleshooting section.

- If you get a **Connection refused** error, check the connection settings and verify that the server is reachable and there isn't a firewall blocking the traffic to it.

Once you are logged in, you will see only the dashboard page and a toolbar (figure 3).



Figure 3: Main toolbar

The toolbar contains the most frequently used tools. Here is an overview of what can you do with them:















	Search objects with the Query Manager
	Refresh the current view
	Refresh local cache
	Default view for an object. Also, the rack view for rack objects
	Create automation tasks (beta version)
	See the changes made to inventory and application objects
	Manage users and groups
	Change the data model
	Manage how objects can be created inside others
	Create new list types
	Freely design network topologies
	Main tree used to explore physical assets
	Create and manage objects that don't fit in the navigation tree
	Manage client, services and resources associated to them

Table 1: Toolbar items

Data Model Manager

One of the key features of Kuwaiba is that it is completely object-oriented³. It means that every business (Router, City, Port) and application (users, types) element is represented by an **Object** in the application and these objects are in turn product of an reality abstraction called **Class**. Likewise, every attribute is a **Field** in a class. The set of classes, attributes and relationships between them is called data model. There's a default data model, but you can customize it depending on your needs by adding, removing and modifying classes. To achieve this, use the Data Model Manager module (figure 5). The data model is represented as a tree because it's a hierarchical

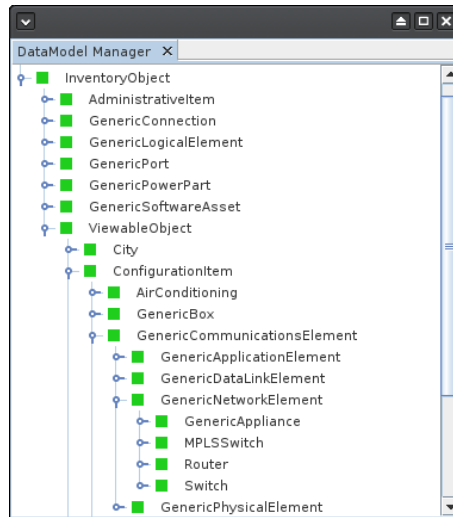


Figure 4: Part of the data model tree

structure. Technically, it's a class hierarchy⁴. The top of the hierarchy (**InventoryObject**) is the most general type of element in the data model and its subclasses represent all the possible elements that will be treated as inventory assets. As you dig deeper into the tree, the classes become more and more specialized and each level inherits the attributes of the parent classes. This kind of structure has two purposes: First, it helps you to organize your classes based on what characteristics they have in common. Secondly, as you will see later in this manual, you can apply operations over top level classes, and they will be propagated to all subclasses. Another root of the data model tree is **GenericObjectList**, and its subclasses are all possible list types (see more details on the subject in the chapter **List Type Manager**).

Important

The **Properties** window allows you to modify the attributes of a selected object in a tree, list or view. If not already open, it's available from the Windows → Properties menu.

The properties of a class can be edited by using the **Properties** window.

³Object-oriented Programming https://en.wikipedia.org/wiki/Object-oriented_programming

⁴Class Hierarchy https://en.wikipedia.org/wiki/Class_hierarchy

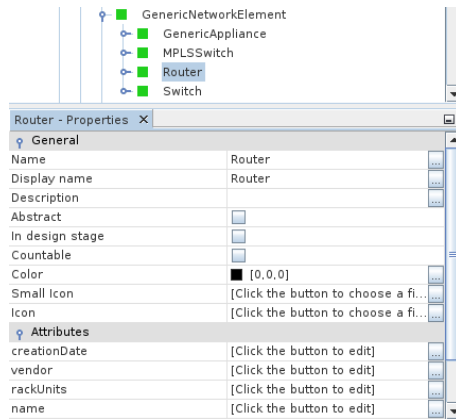


Figure 5: Class **Router** properties

List Type Manager