Workpackage WP26: DBE Portal

Deliverable D26.7: DBE Portal Specification
Version 2

Project funded by the European Community under the “Information Society Technology” Programme
**Contract Number:** 507953  
**Project Acronym:** DBE  
**Title:** Digital Business Ecosystem

**Deliverable N°:** D26.7  
**Due dates:** 30/11/2006  
**Delivery Date:** 30/11/2006

**Short Description:**
A Digital Business Ecosystem Portal is a user friendly entry point to the DBE that provides the means to search, browse and execute DBE Services over the DBE Peer-To-Peer network using nothing more than a web browser. This document sets out the design of the DBE Portal and the requirements to satisfy this design. It also details the implementation of the supporting DBE Portal toolkit and any issues encountered in its implementation.

**Author:** Intel Ireland Ltd.  
**Partners contributed:**  
**Made available to:** Public

**Versioning**

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author, Organisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>24/05/2005</td>
<td>Andy Edmonds, TCD. Initial Draft</td>
</tr>
<tr>
<td>0.2</td>
<td>28/11/2005</td>
<td>Andy Edmonds, Intel Ireland Ltd. Major revision to original initial draft.</td>
</tr>
<tr>
<td>0.3</td>
<td>17/02/2005</td>
<td>Andy Edmonds, Intel Ireland. Final modifications and correction before submittal.</td>
</tr>
<tr>
<td>0.4</td>
<td>28/02/2006</td>
<td>Andy Edmonds, Intel Ireland Ltd. Suggestions from internal reviewer added.</td>
</tr>
<tr>
<td>0.5</td>
<td>09/03/2006</td>
<td>Andy Edmonds, Intel Ireland Ltd. Further suggestions from internal reviewers added.</td>
</tr>
<tr>
<td>0.6</td>
<td>12/11/2006</td>
<td>Andy Edmonds, Intel Ireland Ltd. Additions inserted to reflect 2.0 version of the Portal</td>
</tr>
<tr>
<td>0.7</td>
<td>30/11/2006</td>
<td>Andy Edmonds, Intel Ireland Ltd. Additions inserted based on internal reviewers comments</td>
</tr>
</tbody>
</table>

**Quality check:**

**1st Internal Reviewer:** Pierfranco Ferronato, Soluta.  
**2nd Internal Reviewer:** Juanjo Aparicio, Techideas.
Creative Commons

Attribution-NonCommercial-ShareAlike 2.5

You are free:

- to copy, distribute, display, and perform the work
- to make derivative works

Under the following conditions:

**BY:** Attribution. You must attribute the work in the manner specified by the author or licensor.

**Noncommercial.** You may not use this work for commercial purposes.

**Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.
Table of Contents

1 EXECUTIVE SUMMARY ................................................................. 8

2 INTRODUCTION ................................................................................. 9
  2.1 Background information ............................................................... 9
  2.2 The DBE Portal ............................................................................ 10

3 DBE PORTAL FEATURE REQUIREMENTS ............................................. 12

4 DBE PORTAL USE CASES ................................................................. 15
  4.1 DBE User Use-cases .................................................................. 16
    4.1.1 Use Cases for Further Work ................................................... 17
  4.2 DBE Portal Use-cases ................................................................. 18
  4.3 DBE Provider Use-cases ............................................................. 20
    4.3.1 Use Cases for Further Work ................................................... 21
  4.4 DBE Consumer Use-cases ........................................................... 21
    4.4.1 Use Cases for Further Work ................................................... 22

5 DBE PORTAL CONSIDERATIONS ...................................................... 23
  5.1 DBE Portal Registration ............................................................... 23
  5.2 Users and the DBE Portal ............................................................. 25
  5.3 Security and the DBE Portal ........................................................ 26

6 DBE PORTAL TOOLKIT ................................................................. 28
  6.1 Remote UI Retrieval .................................................................... 29
  6.2 Remote Service Execution .......................................................... 30
  6.3 Portal Toolkit Logical Structure .................................................... 31
  6.4 Portal Toolkit Pending Issues ........................................................ 33

7 DBE PORTAL DEPENDENCIES ...................................................... 34

8 CASE STUDIES ............................................................................... 36
  8.1 Creating a Website Using the Portal ............................................... 36
  8.2 Integrating the Portal into an Existing Website ............................... 38

9 REFERENCES .................................................................................. 39
10 GLOSSARY ........................................................................................................................................... 40

11 APPENDIX ........................................................................................................................................... 41

11.1 Configuring the DBE Portal ............................................................................................................... 41

*JavaDoc for the Portal Toolkit* ................................................................................................................... 41
Index of Figures

Figure 1 An Overview of the DBE Portal ........................................................................................................... 10
Figure 2 DBE Portal and Related Components .......................................................................................... 11
Figure 3 DBE User Use-cases ..................................................................................................................... 16
Figure 4 DBE Portal Use-cases .................................................................................................................. 18
Figure 5 DBE Provider Use-case ................................................................................................................ 20
Figure 6 DBE Consumer Use-case ............................................................................................................. 21
Figure 7 A Single User and a DBE Portal .................................................................................................. 25
Figure 8 Multiple Users with the Same Domain and a DBE Portal ............................................................. 25
Figure 9 Single User Authentication ......................................................................................................... 26
Figure 10 Multiple User Authentications: Scenario 1 ............................................................................. 27
Figure 11 Multiple User Authentication: Scenario 2 ............................................................................ 27
Figure 12 Remote UI Retrieval .................................................................................................................. 29
Figure 13 Remote UI Retrieval ................................................................................................................ 30
Figure 14 DBE Portal Single User Service Invocation ............................................................................ 30
Figure 15 Remote Service Execution ....................................................................................................... 31
Figure 16 DBE Portal Client Class Diagram ............................................................................................ 32
Figure 17 DBE Portal Service Class Diagram .......................................................................................... 33
Figure 18 DBE Portal Interdependencies .................................................................................................. 34
Figure 19 Default Portal Website .............................................................................................................. 36
Figure 20 Customised Portal Website ....................................................................................................... 38
### Index of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Reference Table of Use Cases</td>
<td>15</td>
</tr>
<tr>
<td>Table 2</td>
<td>Find Services Use-case</td>
<td>16</td>
</tr>
<tr>
<td>Table 3</td>
<td>Browse Service Information Use-case</td>
<td>16</td>
</tr>
<tr>
<td>Table 4</td>
<td>Invoke Service UI Use-case</td>
<td>17</td>
</tr>
<tr>
<td>Table 5</td>
<td>Manage Profile Use-case</td>
<td>17</td>
</tr>
<tr>
<td>Table 6</td>
<td>Access DBE Components Use-case</td>
<td>17</td>
</tr>
<tr>
<td>Table 7</td>
<td>Register with Identity Service Use-case</td>
<td>18</td>
</tr>
<tr>
<td>Table 8</td>
<td>Login/Restore Session Use-case</td>
<td>18</td>
</tr>
<tr>
<td>Table 9</td>
<td>Browse DBE Services Use-case</td>
<td>18</td>
</tr>
<tr>
<td>Table 10</td>
<td>Disseminate SME Information Use-case</td>
<td>18</td>
</tr>
<tr>
<td>Table 11</td>
<td>Register With DBE Use-case</td>
<td>19</td>
</tr>
<tr>
<td>Table 12</td>
<td>Administer Service Use-case</td>
<td>20</td>
</tr>
<tr>
<td>Table 13</td>
<td>Deploy Service Use-case</td>
<td>20</td>
</tr>
<tr>
<td>Table 14</td>
<td>Author Service UI Use-case</td>
<td>21</td>
</tr>
<tr>
<td>Table 15</td>
<td>Compose Service Use-case</td>
<td>21</td>
</tr>
<tr>
<td>Table 16</td>
<td>Bookmark Service Use-case</td>
<td>22</td>
</tr>
</tbody>
</table>
1 Executive Summary

A Digital Business Ecosystem Portal is a user friendly entry point to the DBE that provides the means to search, browse and execute DBE Services over the DBE Peer-To-Peer network using nothing more than a web browser. This document sets out the design of the DBE Portal and the requirements to satisfy this design. It also details the implementation of the supporting DBE Portal toolkit and any issues encountered in its implementation.
2 Introduction

This is a report based on design of a means to search and access DBE services in a way inspired by current trends in web applications. It is written as a part of the deliverable “D26.7 DBE Portal Specification Version 2” within the Digital Business Ecosystem project. This deliverable is composed of both this report and supplementing code. The code and implementation that this document refers to can be freely accessed at http://swallow.sf.net and is distributed as part of the ExE [14].

2.1 Background information

From the DBE web site (http://www.digital-ecosystem.org):

What is DBE?

The Digital Business Ecosystem (DBE) is an Internet-based software environment in which business applications can be developed and used. The unique feature of the DBE is that applications within the ecosystem are able to perform new functions that were, up to now, undreamed of by users.

The DBE is an open, free environment where even the smallest specialist software developer can participate competitively in the massive global marketplace for business applications. It will enable end users to easily access and use those applications as services, and to have the benefits of intelligence, interaction and adaptation as the software evolves in response to their own usage and that of others. The initial target of the DBE is those complex commercial transactions and processes that are not easily or economically served by current even state-of-the-art software technologies.
2.2 The DBE Portal

A Digital Business Ecosystem [15] (DBE) Portal is a user friendly entry point to the DBE that provides the means to search, browse and execute DBE Services over the DBE Peer-To-Peer (P2P) network using nothing more than a web browser. Its relationship with the Execution Environment (ExE) is shown in Figure 1. A DBE Portal enables any user of the DBE with a minimum of technical know-how, and without any installation of DBE applications, to access, consume and use the DBE and the services it hosts.

A DBE Portal is an evolved and enhanced DBE Desktop [1]. The initial release of the DBE Portal seeks to match the functionality already contained in the DBE Desktop. The main functionality presented in the DBE Portal, just as in the DBE Desktop, is the ability to search, browse and execute services. The development of the DBE Desktop has halted except for critical bug fixes. Future feature additions and functionality enhancements will be folded into the DBE Portal development effort.

The method of interfacing with the DBE Portal is by using any web browser (e.g. Firefox, Internet Explorer, Mozilla, and Opera). The DBE Portal does not require any heavy client side plug-ins such as Java run-time environments and as such the web browser is the thin client. All computation is performed on the servent, through the DBE Portal, with the results of computation displayed on the client (web browser). The computation on the servent [8] is performed by interfacing with the DBE via the toolkits and frameworks provided by the Java development kit [16] (JDK) and those already developed by DBE partners. These include the servent Application Programming Interfaces (API) and the DBE client toolkit. The framework that allows DBE services to display user interfaces within the client (web browser) is specified in deliverable 20.1 [2].
A programmer’s toolkit, the Portal Toolkit, has been developed to allow the execution of services through user interfaces (UI) displayed by the end-users’ web browsers.

Taking this approach allows any user of the DBE, without an installation of any DBE applications, to access the DBE with just the bare minimum of a web browser.

The DBE Portal is dependent on other DBE components as shown in the diagram below (Figure 2). Where a user does not have the facility to run their own servent and consequently a DBE Portal, it could be possible, where a Small-to-Medium Enterprise (SME) allows it, for that user to use another SME’s DBE Portal to search, browse and execute DBE services. For this to be an attractive proposition for a SME to consider, incentives must be offered to encourage them to allow lone users access their DBE Portal.

![Figure 2 DBE Portal and Related Components](image-url)
3 DBE Portal Feature Requirements

The basic premise of the DBE Portal is to allow any user of the DBE, be they consumers\(^1\) or providers\(^2\), to interact with services advertised in the DBE using the following three basic functionalities:

- **Search** – using the DBE Portal a DBE user can provide search terms that describe the service that the user is looking for. The user can also be presented with the option to formulate a query using advanced query operators. The returned results are services that best match the entered search terms.

- **Browse Service Information** – After submitting a search request and retrieving back the results, the user can view additional information about a service that the user wants to use.

- **Execute** – using the results from the action of searching or browsing, once a suitable service is selected, the user can invoke and execute that service. From the point of view of the DBE Portal, the execution of a service involves the retrieval of the user interface of that service, and displaying it to the requesting user. On entering data, the user can then execute the UI’s functionality.

It is this set of functionality that has been implemented in the first version of the DBE Portal. The above features fall into the category of application/service functionality and are features related to direct manipulation of DBE services. With the necessary features (listed above) implemented, it is envisioned that new and extra functionalities will be added to the DBE Portal, especially those encapsulated by the general notion of account management, which includes tasks such as service bookmarking, profile management, workflow management etc.

Although the DBE Portal has been designed to satisfy these basic and mandatory use cases, additional functionality has also been considered. Although considered important, it should be noted that the additional functionality will only be implemented when the core functionality of the DBE Portal is complete and released. The secondary functionalities (use cases) fall into the categories of infrastructural or informational and include:

- Facilities to provide a free “web-presence” of the SME not only to users within the DBE network but also to the wider population of users that use the World Wide Web (WWW). This will provide

\(^1\) A consumer can be viewed as a classic client; an entity that uses the functionality of a service to achieve an end-goal.

\(^2\) A provider, in the sense of DBE, is an entity that provides a service that can be consumed. The service that the provider offers may utilise external services and in this provider specific context, the provider is also a consumer too.
a means for the SME to disseminate information about itself and its services. To this end, a Content Management System (CMS) for the organisation of SME information may be used. A CMS is a system used to organize and facilitate collaborative content creation. It provides a simple way to create, maintain and update content hosted on a DBE Portal and display it using the familiar mechanisms of a web server via a “web site”.

- Using a DBE Portal, a DBE user could, starting from a particular taxonomy, drill down until a service meeting the user’s requirements is discovered. The desired functionality of browsing DBE service could emulate that of Universal Description, Discovery and Integration [13] (UDDI) service listing, for example:
  - White directory: based on address and contact information. This information can be found within a service’s Business Modelling Language (BML) data model (M0), contained in the service’s service manifest.
  - Yellow directory: based on service categorised by industry/business. This information can be found within a service’s BML model (M1), contained in the service’s service manifest.
  - Green directory: based on technical information about services. This type of information can be found within a service’s Service Definition Language (SDL) model, contained in the service’s service manifest.

- A means of user registration. This will allow registration of new identities with the DBE using the Identity Service (IS) as it is anticipated that DBE Portals will be entry points (i.e. registration agents) for an SME wishing to join the DBE. It will be here where the SME will register with the DBE in order to provide its services.

- A way to allow user profile creation and management. This should facilitate the modification and management of a registered user’s profile [3] identified by the user’s IS identity. Using the user’s profile, it is hoped that preferred services can be bookmarked and stored by saving a reference to the service’s Service Manifest Identifier (SMID). It is foreseen that the Distributed Storage System (DSS) could be used as a mechanism to allow the storage of such information and the user’s profile.

- A DBE Portal should also be registered with a directory-type component (see Ch. 5) when the SME starts its servent and comes online. Conversely, it should be unregistered when the SMEs servent is shutdown and goes offline. This task may also involve updating of the DBE Portal’s endpoint information with the directory-type component especially if the SME’s Internet Service
Provider (ISP) mandates the use of Dynamic Host Control Protocol (DHCP) [10] allocated Internet Protocol (IP) [11] addresses and or the DBE Portal is located behind a Network Address Translator (NAT) [12].

- A distributed DBE service development environment where DBE developers can create service compositions and deploy those compositions. User interfaces could, potentially, also be created within a DBE Portal to visually represent those service compositions to consumers.

- A means and mechanism to access the DBE applications. The DBE suite of applications enables the development of DBE services, the consumption of services and the provisioning of DBE services. These applications include the DBE Studio and related plugins, the DBE Servent and the DBE Evolutionary Environment (EvE). Also any frameworks that a developer may be interested in using e.g. the APA should be accessible.

- A place where DBE related statistics can be viewed, e.g., number of Knowledge Base (KB) nodes, number of currently “live” SMEs, total number of “live” DBE services, network topology and it size. This task could be performed by another DBE core service that DBE Portal could use, which could perhaps supply greater information than the partial view of the network available to the DBE Portal. This service could track various P2P system nodes and topologies formed between servent and P2P system nodes. Another, but simpler way to get some statistics would be to issue a query to the Semantic Registry (SR) requesting how many entities of the DBE Portal type are currently registered.

- Support for the resumption of DBE user sessions could also be included in the portal so that DBE users could resume work from where they last left after logging out of DBE. By supporting this, it would enable the retrieval of results from long running transactions as it is infeasible to expect DBE users to remain logged in to the DBE over period of days. The user’s state could be saved in DSS.
4 DBE Portal Use Cases

What follows is a listing of possible use cases related to the DBE Portal. Those listed below may not be implemented in the initial release of the DBE Portal but will start to form a development road map for the DBE Portal. From the DBE technical annex, it is stated that there will be two releases, both deliverables, of the DBE Portal corresponding to the versions of DBE Portal Version 1 and DBE Portal Version 2. Below is a table summarising the use cases collected so far for the DBE Portal (those use cases in *italics* are suggested future work).

<table>
<thead>
<tr>
<th>Use Case Reference</th>
<th>Version Implemented</th>
<th>Use Case Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UUC1</td>
<td>0.1</td>
<td>Find Services Use-case</td>
</tr>
<tr>
<td>UUC2</td>
<td>0.1</td>
<td>Browse Service Information Use-case</td>
</tr>
<tr>
<td>UUC3</td>
<td>0.1</td>
<td>Invoke Service UI Use-case</td>
</tr>
<tr>
<td>UUC4</td>
<td>0.2</td>
<td>Manage Profile Use-case</td>
</tr>
<tr>
<td>UUC5</td>
<td>0.1</td>
<td>Access DBE Components Use-case</td>
</tr>
<tr>
<td>UUC6</td>
<td>n/a</td>
<td>Register with Identity Service Use-case</td>
</tr>
<tr>
<td>UUC7</td>
<td>n/a</td>
<td>Login/Restore Session Use-case</td>
</tr>
<tr>
<td>UUC8</td>
<td>n/a</td>
<td>Browse DBE Services Use-case</td>
</tr>
<tr>
<td>PoUC1</td>
<td>0.1</td>
<td>Disseminate SME Information Use-case</td>
</tr>
<tr>
<td>PoUC2</td>
<td>0.2</td>
<td>Register With DBE Use-case</td>
</tr>
<tr>
<td>PrUC1</td>
<td>0.2</td>
<td>Administer Service Use-case</td>
</tr>
<tr>
<td>PrUC2</td>
<td>0.2</td>
<td>Deploy Service Use-case</td>
</tr>
<tr>
<td>PrUC3</td>
<td>n/a</td>
<td>Author Service UI Use-case</td>
</tr>
<tr>
<td>PrUC4</td>
<td>n/a</td>
<td>Compose Service Use-case</td>
</tr>
<tr>
<td>CUC1</td>
<td>n/a</td>
<td>Bookmark Service Use-case</td>
</tr>
</tbody>
</table>

*Table 1 Reference Table of Use Cases*
4.1 **DBE User Use-cases**

![Diagram of DBE User Use-cases]

**Use Case UUC1: Find Service**
- **Area**: DBE Portal
- **Objective**: Find existing DBE Service through a Web Browser
- **Actors**: User
- **Pre-Conditions**: DBE Services must be available
- **Post-Conditions**: The user will issue search queries in a Web Browser in order to find services he/she is interested in.
- **Dependencies**: User Interface (web-based), Semantic Registry
- **Release**: Version 1
- **Relevant Partner**: TUC, SUN, Intel

**Table 2 Find Services Use-case**

<table>
<thead>
<tr>
<th>Use Case UUC1</th>
<th>Find Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Find existing DBE Service through a Web Browser</td>
</tr>
<tr>
<td>Actors</td>
<td>User</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td>DBE Services must be available</td>
</tr>
<tr>
<td>Post-Conditions</td>
<td>The user will issue search queries in a Web Browser in order to find services he/she is interested in.</td>
</tr>
<tr>
<td>Dependencies</td>
<td>User Interface (web-based), Semantic Registry</td>
</tr>
<tr>
<td>Release</td>
<td>Version 1</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>TUC, SUN, Intel</td>
</tr>
</tbody>
</table>

**Use Case UUC2: Browse Service Information**
- **Area**: DBE Portal
- **Objective**: Browse DBE Service Information through a Web Browser
- **Actors**: User
- **Pre-Conditions**: DBE Services must be available
- **Post-Conditions**: Semantic Registry, Portal Toolkit
- **Release**: Version 1
- **Relevant Partner**: TUC, Intel

**Table 3 Browse Service Information Use-case**

<table>
<thead>
<tr>
<th>Use Case UUC2</th>
<th>Browse Service Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Browse DBE Service Information through a Web Browser</td>
</tr>
<tr>
<td>Actors</td>
<td>User</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td>DBE Services must be available</td>
</tr>
<tr>
<td>Post-Conditions</td>
<td>Semantic Registry, Portal Toolkit</td>
</tr>
<tr>
<td>Release</td>
<td>Version 1</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>TUC, Intel</td>
</tr>
</tbody>
</table>
### Use Case UUC3: Invoke Service UI

**Area:** DBE Portal  
**Objective:** Use an existing DBE Service’s UI through a Web Browser  
**Actors:** User  
**Pre-Conditions:** DBE Services must be available  
The search query must return at least two DBE Services  
**Post-Conditions:**  
**Description:** The user will use an existing DBE Service through a Web Browser. The service will display an OpenLaszlo user interface with input fields to fill in by the user.  
**Dependencies:** Web-based GUI Development Tool, Portal Toolkit  
**Release:** Version 1  
**Relevant Partner:** Intel, TUC, SUN  

*Table 4 Invoke Service UI Use-case*

### Use Case UUC4: Manage Profile

**Area:** DBE Portal  
**Objective:** Edit, update and manage a user’s DBE preferences.  
**Actors:** User  
**Pre-Conditions:** User is logged in. User-profile service is available  
**Post-Conditions:**  
**Description:** The user can manage his/her preferences and persist these to suitable storage.  
**Dependencies:** User profiling service  
**Release:** Version 2  
**Relevant Partner:** FZI, Intel  

*Table 5 Manage Profile Use-case*

### Use Case UUC5: Access DBE Components

**Area:** DBE Portal  
**Objective:** Download DBE components to develop and access DBE services  
**Actors:** User  
**Pre-Conditions:**  
**Post-Conditions:**  
**Description:** A user wanting to develop DBE services can access the necessary resources to accomplish this.  
**Dependencies:** SourceForge project site  
**Release:** Version 1  
**Relevant Partner:** Intel  

*Table 6 Access DBE Components Use-case*

### 4.1.1 Use Cases for Further Work

### Use Case UUC6: Register with Identity Service

**Area:** DBE Portal  
**Objective:** Issue a registration request for a DBE Identity and receive corresponding credentials  
**Actors:** User
Pre-Conditions | User is not registered with the identity service  
Post-Conditions | User has a DBE identity  
Description | A user can register for a DBE Identity though the DBE Portal  
Dependencies | Identity service.  
Relevant Partner | TCD, Intel  

Table 7 Register with Identity Service Use-case

<table>
<thead>
<tr>
<th>Use Case UUC7</th>
<th>Login/Restore Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Return to a long-running session through the DBE Portal</td>
</tr>
<tr>
<td>Actors</td>
<td>User</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td>An existing session running</td>
</tr>
<tr>
<td>Post-Conditions</td>
<td>When a user executes a long-running process e.g. a workflow, the user will be able to log out and at a later stage log back in and check the status of the process</td>
</tr>
<tr>
<td>Dependencies</td>
<td>Identity service, user-profiling service</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>University of Surrey, Intel</td>
</tr>
</tbody>
</table>

Table 8 Login/Restore Session Use-case

<table>
<thead>
<tr>
<th>Use Case UUC8</th>
<th>Browse DBE Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Browse DBE Service Information through a Web Browser in a UDDI fashion</td>
</tr>
<tr>
<td>Actors</td>
<td>User</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td>DBE Services must be available</td>
</tr>
<tr>
<td>Post-Conditions</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>Dependencies</td>
<td>Semantic Registry, Portal Toolkit</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>TUC, Intel</td>
</tr>
</tbody>
</table>

Table 9 Browse DBE Services Use-case

### 4.2 DBE Portal Use-cases

![Figure 4 DBE Portal Use-cases](image-url)
<table>
<thead>
<tr>
<th>Use Case PoUC1</th>
<th>Disseminate SME Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Provide an “out of the box” web site.</td>
</tr>
<tr>
<td>Actors</td>
<td>Portal</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td></td>
</tr>
<tr>
<td>Post-Conditions</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>This will provide a free web presence that will allow SMEs provide more information about its business.</td>
</tr>
<tr>
<td>Dependencies</td>
<td>Content Management System (Possibly)</td>
</tr>
<tr>
<td>Release</td>
<td>Version 1</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>Intel</td>
</tr>
</tbody>
</table>

Table 10 Disseminate SME Information Use-case

<table>
<thead>
<tr>
<th>Use Case PoUC2</th>
<th>Register With DBE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Register the Portal with DBE infrastructure</td>
</tr>
<tr>
<td>Actors</td>
<td>Portal</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td></td>
</tr>
<tr>
<td>Post-Conditions</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>By registering the Portal with DBE infrastructure, it will be possible to search for Portals within a particular business domain.</td>
</tr>
<tr>
<td>Dependencies</td>
<td>KB, SR</td>
</tr>
<tr>
<td>Release</td>
<td>Version 2</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>Intel, SUN</td>
</tr>
</tbody>
</table>

Table 11 Register With DBE Use-case
### 4.3 DBE Provider Use-cases

![Diagram of DBE Provider Use-cases](image)

**Figure 5 DBE Provider Use-case**

<table>
<thead>
<tr>
<th>Use Case PrUC1</th>
<th>Administer Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Area</strong></td>
<td>Portal</td>
</tr>
<tr>
<td><strong>Objective</strong></td>
<td>Administer various services of the ExE</td>
</tr>
<tr>
<td><strong>Actors</strong></td>
<td>Provider</td>
</tr>
<tr>
<td><strong>Pre-Conditions</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Post-Conditions</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>This will allow the administration of the various components of the ExE through the portal</td>
</tr>
<tr>
<td><strong>Dependencies</strong></td>
<td>P2P system, Servent, KB, SR</td>
</tr>
<tr>
<td><strong>Release</strong></td>
<td>Version 2</td>
</tr>
<tr>
<td><strong>Relevant Partner</strong></td>
<td>Intel, SUN</td>
</tr>
</tbody>
</table>

*Table 12 Administer Service Use-case*

<table>
<thead>
<tr>
<th>Use Case PrUC2</th>
<th>Deploy Service</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Area</strong></td>
<td>Portal</td>
</tr>
<tr>
<td><strong>Objective</strong></td>
<td>Deploy an implemented service</td>
</tr>
<tr>
<td><strong>Actors</strong></td>
<td>Provider</td>
</tr>
<tr>
<td><strong>Pre-Conditions</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Post-Conditions</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>This will allow the deployment of an implemented service through the Portal</td>
</tr>
<tr>
<td><strong>Dependencies</strong></td>
<td>Servent</td>
</tr>
<tr>
<td><strong>Release</strong></td>
<td>Version 2</td>
</tr>
<tr>
<td><strong>Relevant Partner</strong></td>
<td>Intel, SUN</td>
</tr>
</tbody>
</table>

*Table 13 Deploy Service Use-case*
4.3.1 Use Cases for Further Work

<table>
<thead>
<tr>
<th>Use Case PrUC3</th>
<th>Author Service UI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Allow users create declarative UIs within the Portal</td>
</tr>
<tr>
<td>Actors</td>
<td>Provider</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td></td>
</tr>
<tr>
<td>Post-Conditions</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>A user can use the Portal to author declarative UIs and associating it with a service on the portal’s servant</td>
</tr>
<tr>
<td>Dependencies</td>
<td>UI Technology</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>UCE, SUN, Intel</td>
</tr>
</tbody>
</table>

*Table 14 Author Service UI Use-case*

<table>
<thead>
<tr>
<th>Use Case PrUC4</th>
<th>Compose Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Allows users to create service compositions</td>
</tr>
<tr>
<td>Actors</td>
<td>Provider</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td></td>
</tr>
<tr>
<td>Post-Conditions</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>A user will be able to create service compositions of services found within the DBE within the Portal</td>
</tr>
<tr>
<td>Dependencies</td>
<td>Service composition engine</td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>TCD, Intel</td>
</tr>
</tbody>
</table>

*Table 15 Compose Service Use-case*

4.4 DBE Consumer Use-cases

*Figure 6 DBE Consumer Use-case*
4.4.1 Use Cases for Further Work

<table>
<thead>
<tr>
<th>Use Case CUC1</th>
<th>Bookmark Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>DBE Portal</td>
</tr>
<tr>
<td>Objective</td>
<td>Store pointers to used DBE service instances</td>
</tr>
<tr>
<td>Actors</td>
<td>Consumer</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td>Existing services</td>
</tr>
<tr>
<td>Post-Conditions</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>This will allow users store pointers to used DBE services much like a user would bookmark websites</td>
</tr>
<tr>
<td>Dependencies</td>
<td></td>
</tr>
<tr>
<td>Relevant Partner</td>
<td>Intel, TUC</td>
</tr>
</tbody>
</table>

*Table 16 Bookmark Service Use-case*
5 DBE Portal Considerations

5.1 DBE Portal Registration

Originally in the initial stages of the design and specification of the DBE Portal, it was conceived that there would be a DBE Directory, a type of meta-portal named as the DBE Gateway Portal. It would be here where all DBE Portals hosted on servents would register themselves along with their endpoint information. It would be a directory that aggregated all SMEs currently online and organised them according to their ontology.

Although this approach would suffice for searching for DBE Portals and the services hosted within them, it would unfortunately be deficient. The primary reason to reject such a design is that in essence the DBE Gateway Portal would be a single point of failure in an otherwise fully decentralised design. Another major point of rejection is that parallel tracks of development would be necessitated to develop such a Gateway Portal. From exploratory work carried out by the author, it was viewed that a new lightweight directory service would be required to enable the Gateway Portal within DBE. Obviously this would circumvent established frameworks and core services already available within the DBE ecosystem and from a managerial point of view with regards to resource allocation would be inefficient and wasteful. Posed with these problems, it was necessary to find a solution using the current frameworks and DBE core services so that DBE Portals could register themselves within the DBE networks in order to be searchable.

To this end, the notion of modelling a DBE Portal as a DBE service was conceived. What is required for this solution is to describe the DBE Portal as a BML model, a model that should not change from SME to SME. Such a BML model is a model template. The BML template provides a means to search for a group of DBE Portals that fall under a particular type of business domain in a distributed P2P fashion. All that will change is the BML data associated with that model. The BML data required for a DBE Portal will be supplied when it is installed along with the servent and will be a one-time process.

By using features of BML, it is possible to organise SME DBE Portals into categories much in the same way as organisation is performed in folksonomies [4] by using the data modelled in the BML. By following such an approach it is feasible to imagine self-organising structure and categorisation within the DBE where more accurate terms describing a particular business domain are popularised. Those popularised are the terms that are the “fittest” for describing such a domain. It is important to note that by following this approach it does not require that an administrator look after a particular ontology and it is possible to organise services in a navigable, automatically created structure.

Although, initially it made sense to store endpoint information in the BML data, this after further thought was considered infeasible. It would only work where it can be guaranteed that for all SMEs their
endpoint information was immutable and never changing, but for many SMEs this is not the case (e.g. where there IP address is allocated by DHCP). As BML data is considered to be slowly changing over time, it is not a suitable place to store endpoint information, which could be potentially updated every hour. To resolve this, a simple SDL model of a DBE Portal is required. It will be this SDL model and its corresponding interface and methods that will return back endpoint information to locate a SME’s DBE Portal. The interface of the SDL model, shown as a Java interface for brevity, is modelled as follows:

```
<<interface>>>
PortalService
+
+ getUIURL():String
+ getURL():String
```

Snippet 1 DBE Portal SDL Interface

With the DBE Portal models created and implemented the DBE Portal service would be exported to the servent where it will be published (with its service manifest) to the Semantic Registry (SR) and have its service proxy registered with the P2P network. With this performed the DBE Portals contained within the DBE network are then searchable using the existing DBE infrastructure.

Typically, when a SME first wants to log onto the DBE, the SME starts the servent which in turn starts the DBE Portal. As the servent starts, it registers all services that are currently deployed on it including the DBE Portal service. As it registers the DBE Portal service, the servent should also update the DBE Portal’s endpoint information by checking for:

- Renewed DHCP IP address – whether or not the SMEs ISP has requested that the SME’s machine through which it accesses the network should renew its IP address.
- NAT Status – whether or not the servent is behind a NAT and if so its type and the external IP address the servent is contactable at. This NAT information could possibly be obtained by using a STUN [5] server.

Once the DBE Portal service has started up, the DBE Portal to which it corresponds is available through the established mechanisms of discovery and consumption in the DBE.

- Discovery is performed by the SR and consumption of the DBE Portal service is executed by the servent.
- *Consumption* happens when the DBE Portal service’s proxy returns the endpoint of the DBE Portal where it can be located and displayed within a standard web browser.

When the SME shuts down the server, its DBE Portal goes off-line and the service is shutdown thus removing it from the P2P system and making it unavailable. This validates the metaphor “the SME is closed for business” and although closed for business the SME still has a visibility within the DBE, just like a closed shop has on a street.

### 5.2 Users and the DBE Portal

There are a number of scenarios in which users of a service can interact with the DBE Portal and its related services. The simplest case is that of a user, acting as a single entity, directly accessing a DBE Portal that is externally accessible and allows both anonymous and authenticated users interact with it. This is shown in Figure 7.

![Figure 7 A Single User and a DBE Portal](image)

A second case is where multiple users belonging to the same organisation or SME interact with their instance of a DBE Portal. In this case, users within the SME’s domain are the only users interacting with it. This DBE Portal in this case is not accessible from domains outside of the SME’s domain. This is shown in Figure 8.

![Figure 8 Multiple Users with the Same Domain and a DBE Portal](image)
5.3 Security and the DBE Portal

This is related to the previous section on the topic of users and the DBE Portal. What is in question here is where and at what level is authentication and authorisation performed?

In the first case, the single user entity scenario, the single user remains anonymous, or authenticates against the DBE Portal. With the authentication information, it can then, on the behalf of the user, authenticate against the Identity Service (IS). If the DBE Portal allows for anonymous access, then the policies of what an anonymous user can do at that DBE Portal could be set by using the administration interface of it. Below, an example of a single user invoking a service with authentication is shown:

![Figure 9 Single User Authentication]

The next scenario is when there are multiple users within the same SME domain that require the use of services that reside outside of the SME’s domain. The DBE Portal in this case is not accessible from domains outside of the SME’s domain. There are two ways authentication can be performed in this scenario. The first method in which authentication can be performed is shown in Figure 10. In this scenario, the DBE Portal holds the SME’s DBE credentials that allow users invoke services through it. It authenticates for the SME domain that it is part of. Any users within this domain can then use the services advertised within the DBE as it is assumed that they are part of the SME. Taking this approach allows for transparent authentication, makes the implementation of clients easier as they need not implement authentication mechanisms and allows the SME itself manage each of its users independently without having to go to a third party to modify a users permissions or access rights.
The second way authentication can be performed with multiple users within a SME’s domain is shown in Figure 11. In this scenario, every user within the SME’s domain must authenticate against the SME’s DBE Portal. It then authenticates on behalf of the user with the IS. Once authenticated, a user then can use the services within the DBE. Using this scenario allows for each user within the SME to have a stronger identity as each user will have an identity within the IS. This disables a masquerading attack where a hostile user initiates an attack of some sort. This attack would be possible if authentication is performed as in the previous scenario.
6 DBE Portal Toolkit

The DBE Portal Toolkit has been designed to support the:

- Retrieval of a remote UI on a remote servent and display of it using the local servent
- Asynchronous\(^3\) remote execution of a service through a locally hosted user interface

Both the above tasks are carried out by the DBE Portal Toolkit that comprises of:

- **DBE Portal Client** – This component is responsible for the retrieval of UIs and execution of services represented by those UIs. This is a component available within the OpenLaszlo engine, which runs on the Servent.
- **DBE Portal Service**. – This component is responsible for the provisioning of UIs when a request is received for a service’s UI. This is a service written as a DBE Service that runs inside the Servent.

To retrieve the remote UI, the method, `IDBEPortal::getUI()`, contained in the DBE Portal Client is used to retrieve the requested UI. This method returns back the URL of where the UI can be retrieved.

The remote execution of the service through the UI is not UI-technology specific. However, as a *de facto* UI technology the computing domain partners have chosen OpenLaszlo [6] as the UI technology on the recommendation of [2]. There are two main phases related to remote execution of a service when OpenLaszlo is considered.

- **Execution between the UI and servent:** this accomplished using OpenLaszlo-specific JavaRPC [7] that provides an execution binding between the OpenLaszlo engine and the servent.
- **Asynchronous execution between the servent and service:** this is performed using standard servent and APA API’s.

These two tasks of remote UI retrieval and remote service execution are detailed further in the following sections (6.1, 6.2).

---

\(^3\) The invocation is asynchronous between client UI and servent. However execution is not fully asynchronous due to the current P2P system implementation between servents and services.
6.1 Remote UI Retrieval

For the retrieval of a user interface to occur the following steps (illustrated in Figure 12 and Figure 13) take place:

1. The user searches for a service using the Query Formulator-Semantic Discovery Tool (QF-SDT)\(^4\) that is displayed in the user’s web browser. The service could be hosted on a remote servent.
2. The QF-SDT served by portal and displayed by the user’s web browser takes the request and searches the Semantic Registry (SR) for matching services using the Portal Toolkit (PT).
3. A list of matching service offerings (Service Manifests (SM)) are displayed via the QF-SDT interface.
4. The user selects a service interesting him/her.
5. The PT downloads the service proxy on the local servent corresponding to the user selected service.
6. The PT retrieves the service’s UI using the service’s proxy.
7. This UI is mirrored at the local portal using the PT.
8. The QF-SDT is returned a URL to the mirrored UI on the local portal and displays that UI in the user’s web browser using the QF-SDT.

\(^4\) The Query Formulator/ Semantic Discovery Tool mentioned here is an Openlaszlo port of the DBE Studio QF-SDT eclipse plugin. Both have been implemented by TUC.
6.2 Remote Service Execution

From what was shown in 5.2, it can be shown that either the user authenticates themselves to the IS and then directly invokes a particular service or the DBE Portal authenticates on behalf of the user within a SME domain and relays service invocations to and from the user. The first case is typically where a user of the DBE does not need or cannot run the DBE applications. In this case, another means of interacting with the DBE needs to be provided. Figure 14 is an overview of how this process of a single user invoking a service may operate:

![Figure 14 DBE Portal Single User Service Invocation](image1)

The task of executing a service becomes more complicated when authentication is performed on behalf of a user by the DBE Portal. In this case, it is necessary for the DBE Portal to maintain a state table of all invocations made from the authenticated users. The DBE Portal then becomes a service relay. This

![Figure 13 Remote UI Retrieval](image2)
functionality can then be likened to that of a NAT. Within the Portal Toolkit, service execution through a UI is performed in the following fashion and illustrated in Figure 15:

1. The user has a UI to interact with. This UI was acquired from the remote UI retrieval process (6.1).
2. The user presses a button on the UI which calls the PT with:
   a. SMID
   b. Target method name
   c. Method parameters
3. This invocation is passed on to the OpenLaszlo engine on the local servent via JavaRPC to a PT component within the OpenLaszlo engine.
4. To create the remote call to the service by the PT, the PT needs to discover what are the parameter types – this is done by retrieving and reading the SDL types contained in the SM.
5. The PT invokes the Client Side Servent (CSS)
6. The CSS then invokes the remote service using servent infrastructure and the corresponding values returned to the OpenLaszlo client UI via the same JavaRPC mechanism.

![Figure 15 Remote Service Execution](image)

### 6.3 Portal Toolkit Logical Structure

The portal toolkit is split into two logical parts. The first component is one which sits inside the OpenLaszlo engine, the DBE Portal Client. This component is shown below in the following Unified Modelling Language (UML) class diagram:
The second logical part of the DBE Portal Toolkit is the DBE Portal service that runs as a service on the Servent. Its UML class diagram is shown as follows:
6.4 Portal Toolkit Pending Issues

During the implementation and testing of the DBE Portal Toolkit an issue [9] of poor latency and performance was experienced. The symptom noted initially was service UI’s timing out when RPC calls were executed to the backend service associated with the UI. On further investigation, it was found that a synchronous call to the P2P system (Federated Advanced Directory Architecture, FADA) API was causing a bottleneck and was impacted by the propagation delay when P2P system nodes propagate search requests for services. This issue has been solved by the inclusion of asynchronous searches in the implementation of FADA 5.3.5. Service proxy caching was also introduced into the implementation of the Portal Toolkit and this reduces the number of searches on the FADA network and in doing so reduces latency.
7 DBE Portal Dependencies

There are a number of components external to the DBE Portal on which it is dependent on. Synchronisation is required between the DBE Portal and these to ensure an approach that is well integrated and uses each component to its maximum potential and hopefully beyond. The components on which the DBE Portal is dependent (illustrated in Figure 18) are as follows:

- **IS** – In order for the DBE Portal to provide identity registration, validation and management, the Portal will require interfaces to the IS. The partner responsible for this component is TCD.

- **Servent** – This component will be required in order to provide to the DBE Portal information regarding its IP address visibility. The DBE Portal may also need the activation of the servent’s underlying servlet engine that provides for servlets and Java Server Pages (JSP). The partner responsible for this component is SUN.

- **User profiling** – If supported by this component, DBE user preferences pertaining to the DBE Portal can be saved to this service. The partner responsible for this component is FZI.

- **“Yellow pages”** – This is intended to provide a business directory type interface for DBE service. The partner responsible for this component is Soluta.
• Interaction forms – Interaction forms will provide template UIs that a SME can customise if the SME does have the required skills to author a UI on their own behalf. The partner responsible for this component is Soluta.

• Query Formulator/Semantic Discovery Tool – This component will provide the means to search and browse the contents of the DBE network for services. An Openlaszlo version of it has been integrated into the portal. The partner responsible for this component is TUC.

• Transactional Workflow Manager – This component’s interfaces will be required if long running transactions and notification of their completion is to be supported in the DBE Portal. The partner responsible for this component is University of Surrey.
8 Case Studies

In this section two common case studies are presented that SME’s looking to either customise or integrate the default and generic portal may find useful. The first looks at creating a web site for a SME using the generic portal html interface that is supplied with a default installation of the portal. The second case looks at the case where an SME with an existing web site wishes to integrate the Portal search interface into their website.

8.1 Creating a Website Using the Portal

When a user first browses to the portal web site, web site that they will see is one as shown in the figure below.

![Welcome](image)

To customise the default generic web site template (as shown in Figure 19) that comes with the DBE Portal a user needs to have basic knowledge of HTML. The DBE Portal website's content is held within the directory:

"<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/content"

The content of each HTML file is described in the following table:
These files are styled by a CSS file that is located in the directory:

```
<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/content/style
```

Users with CSS experience can then adjust the style and presentation of the content to suit their needs. Although these files are supplied by default, this fact does not preclude a SME from supplying additional ones.

With the content of the SME’s portal created and edited, the final step in customising the Portal is to edit and modify the index file, located at:

```
<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/index.html
```
This file binds the content HTML files together using frames. An example of a customised portal can be seen in Figure 20 and is included in the Portal distribution.

8.2 Integrating the Portal into an Existing Website

To integrate the DBE Portal into your existing website is a simple case of linking to a HTML file. The HTML file in question is the one located at:

<SERVENT_ROOT>/deploy/core-webapps/webapps/portal/content/search.html

This HTML file loads up the OpenLaszlo search interface where users can execute services returned in the search result page.
9 References


[3] FZI; Del 7.2: “Initial Description of Profiling mechanism design and rationale with respect to one or two use cases”. Available from http://www.digital-ecosystem.org/.


[8] DBE Servent; http://swallow.sf.net


[13] Universal Description, Discovery and Integration (UDDI); http://www.uddi.org

[14] Digital Business Ecosystem Execution Environment (ExE); http://swallow.sf.net


[16] SUN Microsystems Java Development Kit (JDK); http://www.java.com
## 10 Glossary

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>APA</td>
<td>Abstract Protocol Adapter.</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface.</td>
</tr>
<tr>
<td>BML</td>
<td>Business Modelling Language.</td>
</tr>
<tr>
<td>CMS</td>
<td>Content Management System.</td>
</tr>
<tr>
<td>CSS</td>
<td>Client Side Servent.</td>
</tr>
<tr>
<td>DBE</td>
<td>Digital Business Ecosystem.</td>
</tr>
<tr>
<td>DIS</td>
<td>Distributed Identity Service.</td>
</tr>
<tr>
<td>DSS</td>
<td>Distributed Storage Service.</td>
</tr>
<tr>
<td>EvE</td>
<td>Evolutionary Environment.</td>
</tr>
<tr>
<td>ExE</td>
<td>Execution Environment.</td>
</tr>
<tr>
<td>FADA</td>
<td>Federated Advanced Directory Architecture.</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol.</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider.</td>
</tr>
<tr>
<td>JDK</td>
<td>Java Development Kit.</td>
</tr>
<tr>
<td>JSP</td>
<td>Java Server Pages.</td>
</tr>
<tr>
<td>KB</td>
<td>Knowledge Base.</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translator.</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer-To-Peer.</td>
</tr>
<tr>
<td>PT</td>
<td>Portal Toolkit.</td>
</tr>
<tr>
<td>QF-SDT</td>
<td>Query Formulator/ Service Discovery Tool.</td>
</tr>
<tr>
<td>SDL</td>
<td>Service Definition Language.</td>
</tr>
<tr>
<td>SM</td>
<td>Service Manifest.</td>
</tr>
<tr>
<td>SME</td>
<td>Small-to-Medium Enterprise.</td>
</tr>
<tr>
<td>SR</td>
<td>Semantic Registry.</td>
</tr>
<tr>
<td>SSS</td>
<td>Server Side Servent.</td>
</tr>
<tr>
<td>TWFM</td>
<td>Transactional Workflow Manager.</td>
</tr>
<tr>
<td>UDDI</td>
<td>Universal Description, Discovery and Integration.</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface.</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modelling Language.</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web.</td>
</tr>
</tbody>
</table>
11 Appendix

11.1 Configuring the DBE Portal

The portal can be configured in two main fashions:

- Through it's web administration interface
- Directly editing its configuration file

For those unfamiliar with configuring the portal, it maybe best to perform the configuration of the portal using the web interface. However, it is recommended that the portal be configured by directly editing the portal’s configuration file.

The web interface to configuring the portal can be accessed at:

```
http://<YOUR_HOSTNAME><YOUR_PORT>/content/config.jsp
```

Here you can select a parameter to change and supply the required value for it. Do note that in order for the changes to occur you will need to restart the servent.

To directly edit the portal’s configuration file you open the file located at:

```
<SERVENT_ROOT>/bin/portalClientConfiguration.properties
```

Then change the required parameters and corresponding values. Those parameters are documented within the configuration file. Again, for the changes to be reflected, the servent needs to be restarted.

JavaDoc for the Portal Toolkit

<table>
<thead>
<tr>
<th>Package Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.dbe.kb.qi.utils</td>
<td>42</td>
</tr>
<tr>
<td>org.dbe.kb.toolkit.proxyutils</td>
<td>44</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal</td>
<td>111</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.client</td>
<td>116</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.execution</td>
<td>124</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.interactionform</td>
<td>129</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.network</td>
<td>131</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.qfsdt</td>
<td>132</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.service</td>
<td>137</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.ui</td>
<td>152</td>
</tr>
<tr>
<td>org.dbe.toolkit.portal.ui.tools</td>
<td>160</td>
</tr>
</tbody>
</table>
public class ServiceDiscoverer
extends Object
Class ServiceDiscoverer

Field Detail

clientCounter

private static long clientCounter

Constructor Detail

ServiceDiscoverer

public ServiceDiscoverer()

Method Detail

getServices

public static Vector getServices(String keywordsAndId)

getModelIds

public static Vector getModelIds(String keywordsAndId)

getSSLModelFromId

public static synchronized Vector getSSLModelFromId(String id)

getBMLModelFromId

public static synchronized Vector getBMLModelFromId(String id)

getSMInfo

public static Vector getSMInfo(String id)

getMoreResults

public static Vector getMoreResults(String sessionId)

performAdvancedSearch

public static Vector performAdvancedSearch(String arguments)

getURLToExecuteService

public static String getURLToExecuteService(String id)
### Class ServiceDiscoverer

**getQuerySessionId**

```java
public static synchronized String getQuerySessionId()
```

**closeSession**

```java
public static void closeSession(String sessionId)
```

**getModelElements**

```java
public static Vector getModelElements(String modelElementParentPath)
```

**getAvailabilityOfService**

```java
public static String getAvailabilityOfService(String id)
```

### Package org.dbe.kb.toolkit.proxyutils

<table>
<thead>
<tr>
<th>Interface Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IRootNode</strong></td>
<td>64</td>
</tr>
<tr>
<td><strong>NameAdapter</strong></td>
<td>73</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AbstractNode</strong></td>
<td>45</td>
</tr>
<tr>
<td>The AbstractNode class is the ancestor of all nodes providing a set of methods that must all implement.</td>
<td></td>
</tr>
<tr>
<td><strong>AttributeNode</strong></td>
<td>47</td>
</tr>
<tr>
<td>The AttributeNode class is the node for the Mof's Attribute classes of the tree control of FormulateQueryWizardPage class.</td>
<td></td>
</tr>
<tr>
<td><strong>BmlInstanceNameAdapter</strong></td>
<td>51</td>
</tr>
<tr>
<td><strong>ContextProvider</strong></td>
<td>52</td>
</tr>
<tr>
<td><strong>ContextRoot</strong></td>
<td>54</td>
</tr>
<tr>
<td>The ContextRoot class is the root of all nodes.</td>
<td></td>
</tr>
<tr>
<td><strong>DefaultNameAdapter</strong></td>
<td>62</td>
</tr>
<tr>
<td><strong>ImmInstanceNameAdapter</strong></td>
<td>63</td>
</tr>
<tr>
<td><strong>Messages</strong></td>
<td>68</td>
</tr>
<tr>
<td><strong>MofHelper</strong></td>
<td>69</td>
</tr>
<tr>
<td>The MofHelper class provides methods for accessing in a unified manner Mof Classes.</td>
<td></td>
</tr>
<tr>
<td><strong>NavigationNode</strong></td>
<td>73</td>
</tr>
<tr>
<td>The NavigationNode class contains a navigable object that has children.</td>
<td></td>
</tr>
<tr>
<td><strong>OdmInstanceNameAdapter</strong></td>
<td>76</td>
</tr>
<tr>
<td><strong>Proxy</strong></td>
<td>77</td>
</tr>
<tr>
<td><strong>ProxyFinding</strong></td>
<td>89</td>
</tr>
<tr>
<td><strong>ProxyHandling</strong></td>
<td>90</td>
</tr>
<tr>
<td>The ProxyHandler class is responsible to connect to a FADA proxy and handle the whole connection and the communication.</td>
<td></td>
</tr>
</tbody>
</table>
The ValueNode class represents a constraint (either hard or soft) on its parent AttributeNode.

Abstract public class **AbstractNode**

extends Object

The AbstractNode class is the ancestor of all nodes providing a set of methods that must all implement. It is a common interface of them in order to be easily explorable. They are nodes of the tree control of FormulateQueryWizardPage class.

**Author:**
George Kotopoulos

**Version:**
1.0

### Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>nameAdapter</td>
<td>NameAdapter</td>
<td>The name Adapter of the object</td>
<td>46</td>
</tr>
<tr>
<td>parent</td>
<td>AbstractNode</td>
<td>The parent of this node</td>
<td>46</td>
</tr>
</tbody>
</table>

### Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbstractNode()</td>
<td></td>
<td>46</td>
</tr>
</tbody>
</table>

### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getChildren()</td>
<td>Object[]</td>
<td>Returns an array with this nodes children.</td>
<td>46</td>
</tr>
<tr>
<td>getId()</td>
<td>String</td>
<td></td>
<td>47</td>
</tr>
<tr>
<td>getName()</td>
<td>String</td>
<td>Gets a String representation of the name of the node.</td>
<td>46</td>
</tr>
<tr>
<td>getObject()</td>
<td>Object</td>
<td>Gets the Object of the current node.</td>
<td>47</td>
</tr>
<tr>
<td>getParent()</td>
<td>AbstractNode</td>
<td>Gets the parent of the current node.</td>
<td>47</td>
</tr>
</tbody>
</table>
### Field Detail

**nameAdapter**

protected `NameAdapter` `nameAdapter`

The name Adapter of the object

---

**parent**

protected `AbstractNode` `parent`

The parent of this node

---

### Constructor Detail

**AbstractNode**

public `AbstractNode`()

---

### Method Detail

**hasChildren**

public abstract boolean `hasChildren()`

Returns true if this node has children. False if it is a leaf node.

**Returns:**

True if this node has children. False if it is a leaf node.

---

**getChildren**

public abstract `Object[]` `getChildren()`

Returns an array with this nodes children.

**Returns:**

An array with this nodes children.

---

**getName**

public `String` `getName()`

Gets a String representation of the name of the node.

**Returns:**

a String representation of the name of the node.
**Class AbstractNode**

**getId**

public String getId()

**getObject**

public abstract Object getObject()

Gets the Object of the current node. Each node is holding an object. Usually a MOF object

**Returns:**

The Object of the current node

**getParent**

public AbstractNode getParent()

Gets the parent of the current node.

**Returns:**

The parent of the current node.

---

**Class AttributeNode**

org.dbe.kb.toolkit.proxyutils

java.lang.Object

\_org.dbe.kb.toolkit.proxyutils.AbstractNode

\_org.dbe.kb.toolkit.proxyutils.AttributeNode

public class AttributeNode extends AbstractNode

The AttributeNode class is the node for the Mof's Attribute classes of the tree control of FormulateQueryWizardPage class.

**Author:**

George Kotopoulos

**Version:**

1.0

---

**Field Summary**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>children</td>
<td>Vector</td>
<td>The childer of this node</td>
<td>48</td>
</tr>
<tr>
<td>object</td>
<td>RefObject</td>
<td>The Mof Attribute object</td>
<td>48</td>
</tr>
<tr>
<td>pathToRoot</td>
<td>Vector</td>
<td>The path to root</td>
<td>49</td>
</tr>
</tbody>
</table>
**Class AttributeNode**

### Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AttributeNode(RefObject node, AbstractNode parent, NameAdapter nameAdapter)</code></td>
<td>Creates a new AttributeNode for a given Mof Attribute.</td>
</tr>
</tbody>
</table>

### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getValue(String op, String val, int weight)</code></td>
<td>Adds a child to this node.</td>
</tr>
<tr>
<td><code>constructPath()</code></td>
<td>Constructs the path to root.</td>
</tr>
<tr>
<td><code>getAttribute()</code></td>
<td>Returns an array with this nodes children.</td>
</tr>
<tr>
<td><code>getObject()</code></td>
<td>Gets the Object of the current node.</td>
</tr>
<tr>
<td><code>getOperations()</code></td>
<td>Gets the available operations for the Attribute object, depending on the type of the object.</td>
</tr>
<tr>
<td><code>getPathToRoot()</code></td>
<td>Gets an iterator containing all the nodes from this one to the root, starting from this one.</td>
</tr>
<tr>
<td><code>getType()</code></td>
<td>Gets the a String representation of the type of the current object.</td>
</tr>
<tr>
<td><code>hasChildren()</code></td>
<td>Returns true if this node has children.</td>
</tr>
<tr>
<td><code>removeValue(ValueNode node)</code></td>
<td>Removes a node from the children of this node.</td>
</tr>
</tbody>
</table>

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode:
- `getChildren`, `getId`, `getName`, `getObject`, `getParent`, `hasChildren`

### Field Detail

- **object**
  - `private RefObject object`  
    The Mof Attribute object

- **children**
  - `private Vector children`  
    The childer of this node
Class **AttributeNode**

**pathToRoot**

private Vector **pathToRoot**

The path to root

### Constructor Detail

**AttributeNode**

```java
public AttributeNode(RefObject node,
                      AbstractNode parent,
                      NameAdapter nameAdapter)
```

Creates a new AttributeNode for a given Mof Attribute.

### Method Detail

**hasChildren**

```java
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

**Overrides:**

`hasChildren` in class `AbstractNode`

**Returns:**

True if this node has children. False if it is a leaf node.

**See Also:**

`org.dbe.dtool.nodes.AbstractNode.hasChildren()`

**getChildren**

```java
public Object[] getChildren()
```

Returns an array with this nodes children.

**Overrides:**

`getChildren` in class `AbstractNode`

**Returns:**

An array with this nodes children.

**See Also:**

`org.dbe.dtool.nodes.AbstractNode.getChildren()`

**getObject**

```java
public Object getObject()
```

Gets the Object of the current node. Each node is holding an Mof Attribute object

**Overrides:**

`getObject` in class `AbstractNode`

**Returns:**

The Mof Attribute object of the current node
**Class AttributeNode**

**See Also:**
org.dbe.dtool.nodes.AbstractNode.getObject()

---

**getType**

```java
public String getType()
```

Gets the a String representation of the type of the current object

---

**xmlSchemaType2OclType**

```java
public static String xmlSchemaType2OclType(String xmlSchemaType)
```

---

**getOperations**

```java
public String[] getOperations()
```

Gets the available operations for the Attribute object, depending on the type of the object.

**Returns:**
The available operations for the Attribute object, depending on the type of the object.

---

**addValue**

```java
public ValueNode addValue(String op, String val, int weight)
```

Adds a child to this node. Constructs a ValueNode object.

**Parameters:**
- `op` - A string representation of the operation
- `val` - The literal value of the constraint
- `weight` - true if it a hard constraint, false otherwise.

**Returns:**
The newly created child ValueNode.

---

**removeValue**

```java
public void removeValue(ValueNode node)
```

Removes a node from the children of this node.

**Parameters:**
- `node` - The ValueNode to be removed.

---

**getPathToRoot**

```java
public Iterator getPathToRoot()
```
Class AttributeNode

Gets an iterator containing all the nodes from this one to the root, starting from this one.

Returns:
An iterator containing all the nodes from this one to the root, starting from this one.

constructPath

private void constructPath()

Constructs the path to root.

Class BmlInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

java.lang.Object

org.dbe.kb.toolkit.proxyutils.BmlInstanceNameAdapter

All Implemented Interfaces:
NameAdapter

public class BmlInstanceNameAdapter
extends Object
implements NameAdapter

Author:
christos

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>BmlInstanceNameAdapter()</td>
<td>51</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getId(RefBaseObject object)</td>
<td>52</td>
</tr>
<tr>
<td>String getName(RefBaseObject object)</td>
<td>52</td>
</tr>
</tbody>
</table>

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter

getId, getName

Constructor Detail

BmlInstanceNameAdapter

public BmlInstanceNameAdapter()
Method Detail

getName

public String getName(RefBaseObject object)

Specified by:
getName in interface NameAdapter

getId

public String getId(RefBaseObject object)

Specified by:
getId in interface NameAdapter

Class ContextProvider

org.dbe.kb.toolkit.proxyutils

java.lang.Object

Lorg.dbe.kb.toolkit.proxyutils.ContextProvider

public class ContextProvider
extends Object

Field Summary

<table>
<thead>
<tr>
<th>private static Vector</th>
<th>BMLContexts</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static Vector</td>
<td>IMMContexts</td>
<td>53</td>
</tr>
<tr>
<td>private static Vector</td>
<td>SDLContexts</td>
<td>53</td>
</tr>
<tr>
<td>private static Vector</td>
<td>SSLContexts</td>
<td>53</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>ConstructorSummary</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContextProvider()</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>MethodSummary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>static String getContextMenuForClass(String className, String metamodel)</td>
<td>53</td>
</tr>
<tr>
<td>Returns the context path that the given class belongs</td>
<td></td>
</tr>
<tr>
<td>static Vector getContextMenuVectorItems(String metamodel)</td>
<td>53</td>
</tr>
<tr>
<td>Returns a vector with the supported contexts for a metamodel</td>
<td></td>
</tr>
<tr>
<td>static Vector getContextMenuPaths(String metamodel)</td>
<td>53</td>
</tr>
<tr>
<td>Returns a vector with the supported contexts paths for a metamodel</td>
<td></td>
</tr>
</tbody>
</table>
Field Detail

BMLContexts

private static Vector BMLContexts

SSLContexts

private static Vector SSLContexts

SDLContexts

private static Vector SDLContexts

IMMContexts

private static Vector IMMContexts

Constructor Detail

ContextProvider

public ContextProvider()

Method Detail

getContextVectorItems

public static Vector getContextVectorItems(String metamodel)

Returns a vector with the supported contexts for a metamodel

Returns:
Vector with supported metamodel contexts

getContextVectorPaths

public static Vector getContextVectorPaths(String metamodel)

Returns a vector with the supported contexts paths for a metamodel

Returns:
Vector with supported metamodel contexts paths

getContextPathForClass

public static String getContextPathForClass(String className, String metamodel)

Returns the context path that the given class belongs
Class ContextProvider

Parameters:

  className - Context's last item name (RefClass name)

Returns:

String representing the context path

Class ContextRoot

org.dbe.kb.toolkit.proxyutils

java.lang.Object

org.dbe.kb.toolkit.proxyutils.AbstractNode

org.dbe.kb.toolkit.proxyutils.ContextRoot

All Implemented Interfaces:

IRootNode

public class ContextRoot
extends AbstractNode
implements IRootNode

The ContextRoot class is the root of all nodes. It represents the query and Mof Class hierarchy of the metamodel. It is the root node of the tree control of FormulateQueryWizardPage class.

Author:

George Kotopoulos

Version:

1.0

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private HashSet&lt;AttributeNodes&gt;</td>
<td></td>
<td>The attribute nodes.</td>
<td>56</td>
</tr>
<tr>
<td>private Object[] children</td>
<td></td>
<td>The children of this node.</td>
<td>56</td>
</tr>
<tr>
<td>private RefObject[] context</td>
<td></td>
<td>The MofClass being the query context</td>
<td>56</td>
</tr>
<tr>
<td>private boolean isInstance</td>
<td></td>
<td></td>
<td>57</td>
</tr>
<tr>
<td>private boolean isSelected</td>
<td></td>
<td></td>
<td>57</td>
</tr>
<tr>
<td>private String metamodel</td>
<td></td>
<td>The metamodel name to which this query applies to</td>
<td>56</td>
</tr>
<tr>
<td>private String name</td>
<td></td>
<td>The query name</td>
<td>57</td>
</tr>
<tr>
<td>private Vector[] path</td>
<td></td>
<td>The package path to the MofClass being query context</td>
<td>56</td>
</tr>
</tbody>
</table>

Fields inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode

ALL, HARD, NONE, SOFT
### Class ContextRoot

#### Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ContextRoot</strong>(Collection root, String name, String metamodel)</td>
<td>57</td>
</tr>
<tr>
<td>Creates a new query, with a name and for a context.</td>
<td></td>
</tr>
<tr>
<td><strong>ContextRoot</strong>(RefBaseObject root, String name, String metamodel)</td>
<td>57</td>
</tr>
<tr>
<td>Creates a new query, with a name and for a context.</td>
<td></td>
</tr>
<tr>
<td><strong>ContextRoot</strong>(RefBaseObject root, String name, QueryNode parent, String metamodel)</td>
<td>57</td>
</tr>
<tr>
<td>Creates a new query, with a name and for a context.</td>
<td></td>
</tr>
</tbody>
</table>

#### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>protected boolean addNode(ValueNode node)</td>
<td>59</td>
</tr>
<tr>
<td>Adds a ValueNode to the query.</td>
<td></td>
</tr>
<tr>
<td>Object[] getChildren()</td>
<td>59</td>
</tr>
<tr>
<td>Returns an array with this nodes children.</td>
<td></td>
</tr>
<tr>
<td>RefObject getContext(int index)</td>
<td>58</td>
</tr>
<tr>
<td>Gets the context of the query.</td>
<td></td>
</tr>
<tr>
<td>String getMetamodel()</td>
<td>61</td>
</tr>
<tr>
<td>Gets a String representation of the query name.</td>
<td></td>
</tr>
<tr>
<td>String getName()</td>
<td>58</td>
</tr>
<tr>
<td>Gets a String representation of the name of the node.</td>
<td></td>
</tr>
<tr>
<td>Iterator getNodes()</td>
<td>60</td>
</tr>
<tr>
<td>Gets an iterator with the ValueNodes of this query.</td>
<td></td>
</tr>
<tr>
<td>Iterator getNodes(boolean isHard)</td>
<td>61</td>
</tr>
<tr>
<td>Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)</td>
<td></td>
</tr>
<tr>
<td>Collection getNodesAsCollection()</td>
<td>61</td>
</tr>
<tr>
<td>Gets an iterator with the ValueNodes of this query.</td>
<td></td>
</tr>
<tr>
<td>protected Vector getNodesAsVector(boolean isHard)</td>
<td>61</td>
</tr>
<tr>
<td>Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)</td>
<td></td>
</tr>
<tr>
<td>int getNodesFlag()</td>
<td>60</td>
</tr>
<tr>
<td>Gets a int flag for the type of the ValueNodes.</td>
<td></td>
</tr>
<tr>
<td>Object getObject()</td>
<td>59</td>
</tr>
<tr>
<td>Gets the Object of the current node.</td>
<td></td>
</tr>
<tr>
<td>Vector getPath()</td>
<td>59</td>
</tr>
<tr>
<td>Gets a Vector containg the package path to the context Mof class of this query.</td>
<td></td>
</tr>
<tr>
<td>Vector getPath(int index)</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>String getQueryName()</td>
<td>58</td>
</tr>
<tr>
<td>Gets a String representation of the query name.</td>
<td></td>
</tr>
<tr>
<td>String getResultType()</td>
<td>58</td>
</tr>
<tr>
<td>Gets the result type of this context, i.e.</td>
<td></td>
</tr>
<tr>
<td>boolean hasChildren()</td>
<td>58</td>
</tr>
<tr>
<td>Returns true if this node has children.</td>
<td></td>
</tr>
<tr>
<td>boolean hasNodes()</td>
<td>60</td>
</tr>
<tr>
<td>Returns true is the query has at least one value node, false otherwise.</td>
<td></td>
</tr>
<tr>
<td>private void init(Collection root, String name, String metamodel)</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>boolean isInstance()</td>
<td>61</td>
</tr>
<tr>
<td>Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)</td>
<td></td>
</tr>
<tr>
<td>boolean isSelected()</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
protected boolean removeNode(ValueNode node)
Remove a ValueNode from the query.

protected void setSelected(boolean val)

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
getChildren, getMetamodel, getNodes, getNodes, getNodesFlag, getPath, getQueryName, getResultType, hasNodes, isInstance

Field Detail

context
private RefObject[] context
The MofClass being the query context

children
private Object[] children
The children of this node. Has always only one, a NavigationNode

path
private Vector[] path
The package path to the MofClass being query context

attributeNodes
private HashSet attributeNodes
The attribute nodes. The actual constraints of this query.

metamodel
private String metamodel
The metamodel name to which this query applies to
Class ContextRoot

name
private String name

The query name

isSelected
private boolean isSelected

isInstance
private boolean isInstance

Constructor Detail

ContextRoot

public ContextRoot(RefBaseObject root,
                   String name,
                   QueryNode parent,
                   String metamodel)

Creates a new query, with a name and for a context.

ContextRoot

public ContextRoot(RefBaseObject root,
                   String name,
                   String metamodel)

Creates a new query, with a name and for a context.

ContextRoot

public ContextRoot(Collection root,
                   String name,
                   String metamodel)

Creates a new query, with a name and for a context.

Method Detail

init

private void init(Collection root,
                   String name,
                   String metamodel)
Class ContextRoot

**getContext**

```java
public RefObject getContext(int index)
```

Gets the context of the query.

**Returns:**

The MofClass context of the query.

**hasChildren**

```java
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

**Overrides:**

`hasChildren` in class `AbstractNode`

**Returns:**

True if this node has children. False if it is a leaf node.

**See Also:**

org.dbe.dtool.nodes.AbstractNode.hasChildren()

**getName**

```java
public String getName()
```

Gets a String representation of the name of the node.

**Overrides:**

`getName` in class `AbstractNode`

**Returns:**

A String representation of the name of the node.

**See Also:**

org.dbe.dtool.nodes.AbstractNode.getName()

**getQueryName**

```java
public String getQueryName()
```

Gets a String representation of the query name.

**Specified by:**

`getQueryName` in interface `IRootNode`

**Returns:**

A String representation of the query name.

**getResultType**

```java
public String getResultType()
```

Gets the result type of this context, i.e. the metamodel.

**Specified by:**

`getResultType` in interface `IRootNode`
**Class ContextRoot**

**Returns:**
The result type of this context, i.e. the metamodel.

---

**getObject**

```java
public Object getObject()
```

Gets the Object of the current node. It is the context and its type is MofClass

**Overrides:**
`getObject` in class `AbstractNode`

**Returns:**
The MofClass of the context

**See Also:**
`org.dbe.dtool.nodes.AbstractNode.getObject()`

---

**getChildren**

```java
public Object[] getChildren()
```

Returns an array with this nodes children.

**Specified by:**
`getChildren` in interface `IRootNode`

**Overrides:**
`getChildren` in class `AbstractNode`

**Returns:**
An array with this nodes children.

**See Also:**
`org.dbe.dtool.nodes.AbstractNode.getChildren()`

---

**getPath**

```java
public Vector getPath()
```

Gets a Vector containing the package path to the context Mof class of this query.

**Specified by:**
`getPath` in interface `IRootNode`

**Returns:**
a Vector containing the package path to the context Mof class of this query.

---

**getPath**

```java
public Vector getPath(int index)
```

---

**addNode**

```java
protected boolean addNode(ValueNode node)
```

Adds a ValueNode to the query.
Class ContextRoot

**Parameters:**
node - The node to be added.

**Returns:**
true if added, false otherwise.

---

**removeNode**

protected boolean removeNode(ValueNode node)

Remove a ValueNode from the query.

**Parameters:**
node - The ValueNode to be removed.

**Returns:**
true if removed successfully, false otherwise.

---

**hasNodes**

public boolean hasNodes()

Returns true is the query has at least one value node, false otherwise.

**Specified by:**
hasNodes in interface IRootNode

**Returns:**
true is the query has at least one value node, false otherwise.

---

**getNodesFlag**

public int getNodesFlag()

Gets a int flag for the type of the ValueNodes. Valid values are NONE (0), HARD (1), SOFT (2) and ALL (3).

**Specified by:**
getNodesFlag in interface IRootNode

**Returns:**
NONE (0), HARD (1), SOFT (2) and ALL (3).

---

**getNodes**

public Iterator getNodes()

Gets an iterator with the ValueNodes of this query.

**Specified by:**
getNodes in interface IRootNode

**Returns:**
The ValueNodes of this query.
**Class ContextRoot**

**getNodesAsCollection**

```java
public Collection getNodesAsCollection()
```

Gets an iterator with the ValueNodes of this query.

**Returns:**
The ValueNodes of this query.

**getNodes**

```java
public Iterator getNodes(boolean isHard)
```

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

**Specified by:**
getNodes in interface IRootNode

**Parameters:**
isHard - if the ValueNodes to be returned are hard constrains or not.

**Returns:**
An Iterator with the ValueNodes of a certain type (hard or soft constrains)

**getNodesAsVector**

```java
protected Vector getNodesAsVector(boolean isHard)
```

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

**Parameters:**
isHard - if the ValueNodes to be returned are hard constrains or not.

**Returns:**
An Iterator with the ValueNodes of a certain type (hard or soft constrains)

**isInstance**

```java
public boolean isInstance()
```

**Description copied from interface: IRootNode**

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

**Specified by:**
isInstance in interface IRootNode

**Returns:**
An Iterator with the ValueNodes of a certain type (hard or soft constrains)

**getMetamodel**

```java
public String getMetamodel()
```

**Description copied from interface: IRootNode**

Gets a String representation of the query name.

**Specified by:**
getMetamodel in interface IRootNode
Returns:
a String representation of the query name.

setSelected

protected void setSelected(boolean val)

isSelected

public boolean isSelected()

Class DefaultNameAdapter

org.dbe.kb.toolkit.proxyutils

java.lang.Object

org.dbe.kb.toolkit.proxyutils.DefaultNameAdapter

All Implemented Interfaces:
   NameAdapter

public class DefaultNameAdapter extends Object implements NameAdapter

Author:
   gkoto

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DefaultNameAdapter()</td>
<td>62</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getId(RefBaseObject object)</td>
<td>63</td>
</tr>
<tr>
<td>String getName(RefBaseObject object)</td>
<td>63</td>
</tr>
</tbody>
</table>

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter

getId, getName

Constructor Detail

DefaultNameAdapter

public DefaultNameAdapter()
Class DefaultNameAdapter

Method Detail

getName

public String getName(RefBaseObject object)

Specified by:

getName in interface NameAdapter

getId

public String getId(RefBaseObject object)

Specified by:

getId in interface NameAdapter

Class ImmInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

java.lang.Object

org.dbe.kb.toolkit.proxyutils.ImmInstanceNameAdapter

All Implemented Interfaces:

NameAdapter

public class ImmInstanceNameAdapter
extends Object
implements NameAdapter

Author:

christos

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImmInstanceNameAdapter()</td>
<td>64</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getId(RefBaseObject object)</td>
<td>64</td>
</tr>
<tr>
<td>String getId(RefBaseObject object, String metamodel)</td>
<td>64</td>
</tr>
<tr>
<td>String getName(RefBaseObject object)</td>
<td>64</td>
</tr>
<tr>
<td>String getName(RefObject object, String modelId, String metamodel, Hashtable hashTable)</td>
<td>64</td>
</tr>
</tbody>
</table>

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter

getId, getName
Class ImmInstanceNameAdapter

Constructor Detail

ImmInstanceNameAdapter

public ImmInstanceNameAdapter()  

Method Detail

getName

public String getName(RefBaseObject object)

Specified by:

getName in interface NameAdapter  

getId

public String getId(RefBaseObject object)

Specified by:

getId in interface NameAdapter  

getId

public String getId(RefBaseObject object,

String metamodel)

getName

public String getName(RefObject object,

String modelId,

String metamodel,

Hashtable hashTable)

Interface IRootNode

org.dbe.kb.toolkit.proxyutils

All Known Implementing Classes:

ContextRoot, QueryNode  

public interface IRootNode

Author:

gkoto
Interface IRootNode

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>int ALL</td>
<td>66</td>
</tr>
<tr>
<td>a int flag for both hard and soft constrains.</td>
<td></td>
</tr>
<tr>
<td>int HARD</td>
<td>65</td>
</tr>
<tr>
<td>a int flag for hard constrains.</td>
<td></td>
</tr>
<tr>
<td>int NONE</td>
<td>66</td>
</tr>
<tr>
<td>a int flag for none constraint.</td>
<td></td>
</tr>
<tr>
<td>int SOFT</td>
<td>65</td>
</tr>
<tr>
<td>a int flag for soft constrains.</td>
<td></td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object[] getChildren()</td>
<td>67</td>
</tr>
<tr>
<td>Returns an array with this nodes children.</td>
<td></td>
</tr>
<tr>
<td>String getMetamodel()</td>
<td>67</td>
</tr>
<tr>
<td>Gets a String representation of the query name.</td>
<td></td>
</tr>
<tr>
<td>Iterator getNodes()</td>
<td>66</td>
</tr>
<tr>
<td>Gets an iterator with the ValueNodes of this query.</td>
<td></td>
</tr>
<tr>
<td>Iterator getNodes(boolean isHard)</td>
<td>67</td>
</tr>
<tr>
<td>Gets an iterator with the ValueNodes of a certain type (hard or soft constrains)</td>
<td></td>
</tr>
<tr>
<td>int getNodesFlag()</td>
<td>66</td>
</tr>
<tr>
<td>Gets a int flag for the type of the ValueNodes.</td>
<td></td>
</tr>
<tr>
<td>Vector getPath()</td>
<td>67</td>
</tr>
<tr>
<td>Gets a Vector containing the package path to the context Mof class of this query.</td>
<td></td>
</tr>
<tr>
<td>String getQueryName()</td>
<td>66</td>
</tr>
<tr>
<td>Gets a String representation of the query name.</td>
<td></td>
</tr>
<tr>
<td>String getResultType()</td>
<td>67</td>
</tr>
<tr>
<td>Gets the result type of this context, i.e.</td>
<td></td>
</tr>
<tr>
<td>boolean hasNodes()</td>
<td>66</td>
</tr>
<tr>
<td>Returns true is the query has at least one value node, false otherwise.</td>
<td></td>
</tr>
<tr>
<td>boolean isInstance()</td>
<td>67</td>
</tr>
<tr>
<td>Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)</td>
<td></td>
</tr>
</tbody>
</table>

Field Detail

HARD

public static final int HARD

a int flag for hard constrains. The value is 1

SOFT

public static final int SOFT

a int flag for soft constrains. The value is 2
public static final int ALL

A int flag for both hard and soft constraints. The value is 3

public static final int NONE

A int flag for none constraint. The value is 0

### Method Detail

**getQueryName**

```java
public String getQueryName()
```

Gets a String representation of the query name.

**Returns:**

A String representation of the query name.

**getNodesFlag**

```java
public int getNodesFlag()
```

Gets an int flag for the type of the ValueNodes. Valid values are NONE (0), HARD (1), SOFT (2) and ALL (3)

**Returns:**

NONE (0), HARD (1), SOFT (2) and ALL (3).

**hasNodes**

```java
public boolean hasNodes()
```

Returns true if the query has at least one value node, false otherwise.

**Returns:**

true if the query has at least one value node, false otherwise.

**getNodes**

```java
public Iterator getNodes()
```

Gets an iterator with the ValueNodes of this query.

**Returns:**

The ValueNodes of this query.
Interface IRootNode

**getNodes**

public Iterator getNodes(boolean isHard)

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

**Parameters:**

- isHard - if the ValueNodes to be returned are hard constrains or not.

**Returns:**

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

**isInstance**

public boolean isInstance()

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

**Returns:**

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

**getMetamodel**

public String getMetamodel()

Gets a String representation of the query name.

**Returns:**

a String representation of the query name.

**getChildren**

public Object[] getChildren()

Returns an array with this nodes children.

**Returns:**

An array with this nodes children.

**See Also:**

org.dbe.dtool.nodes.AbstractNode.getChildren()

**getPath**

public Vector getPath()

Gets a Vector containg the package path to the context Mof class of this query.

**Returns:**

a Vector containg the package path to the context Mof class of this query.

**getResultType**

public String getResultType()

Gets the result type of this context, i.e. the metamodel.

Returns:
The result type of this context, i.e. the metamodel.

Class Messages
org.dbe.kb.toolkit.proxyutils
java.lang.Object

| org.dbe.kb.toolkit.proxyutils.Messages |

public class Messages
extends Object

Field Summary
private static final String BUNDLE_NAME
private static final ResourceBundle RESOURCE_BUNDLE

Constructor Summary
private Messages()

Method Summary
static String getString(String key)

Field Detail

BUNDLE_NAME
private static final String BUNDLE_NAME

RESOURCE_BUNDLE
private static final ResourceBundle RESOURCE_BUNDLE

Constructor Detail

Messages
private Messages()

Method Detail

getString
public static String getString(String key)
public class MofHelper
extends Object

The MofHelper class provides methods for accessing in a unified manner Mof Classes. It uses the JMI package for Mof model and the reflection package.

Author:
George Kotopoulos

Version:
1.0

Field Summary

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static Hashtable hash</td>
<td>70</td>
</tr>
<tr>
<td>A hashtable containing for each type of data its hierarchy</td>
<td></td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>MofHelper()</td>
<td>70</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>static Vector changeToNodes(Vector input, AbstractNode parent, NameAdapter nameAdapter)</td>
<td>72</td>
</tr>
<tr>
<td>Changes Mof classes into Nodes depending on the type of each class</td>
<td></td>
</tr>
<tr>
<td>private static Vector getAllContents(MofClass metaObject)</td>
<td>72</td>
</tr>
<tr>
<td>Gets the features of a MofClass.</td>
<td></td>
</tr>
<tr>
<td>static Vector getAllContentsAsNodes(MofClass metaObject, AbstractNode parent, NameAdapter nameAdapter)</td>
<td>72</td>
</tr>
<tr>
<td>Gets the features of a MofClass as Nodes</td>
<td></td>
</tr>
<tr>
<td>static Vector getFeatures(Attribute metaObject, AbstractNode parent, NameAdapter nameAdapter)</td>
<td>71</td>
</tr>
<tr>
<td>Retrieves all the features of a Mof Attribute (i.e References and Attributes).</td>
<td></td>
</tr>
<tr>
<td>static Vector getFeatures(MofClass metaObject, AbstractNode parent, NameAdapter nameAdapter)</td>
<td>70</td>
</tr>
<tr>
<td>Retrieves all the features of a MofClass (i.e References and Attributes)</td>
<td></td>
</tr>
<tr>
<td>static Vector getFeatures(Reference metaObject, AbstractNode parent, NameAdapter nameAdapter)</td>
<td>70</td>
</tr>
<tr>
<td>Retrieves all the features of a Mof Reference (i.e References and Attributes)</td>
<td></td>
</tr>
<tr>
<td>static Vector getFeatures(RefObject object)</td>
<td>71</td>
</tr>
<tr>
<td>Retrieves all the features of a RefObject (i.e M1 objects).</td>
<td></td>
</tr>
<tr>
<td>static Vector getFeatures(RefObject object, AbstractNode parent, NameAdapter nameAdapter)</td>
<td>71</td>
</tr>
<tr>
<td>Retrieves all the features of a RefObject (i.e M1 objects).</td>
<td></td>
</tr>
<tr>
<td>private static RefObject getObject(RefObject object)</td>
<td>71</td>
</tr>
</tbody>
</table>
private static AbstractNode newNode(RefObject object, AbstractNode parent, NameAdapter nameAdapter)

private static void retrieveHierarchy(String metamodel)

Constructs a hash and retrieves in it the class hierarchies of SSL's Core, Instantiation and Types packages.

private static void retrievePackageHierarchy(RefPackage pack)

Adds to a hash the class hierarchies of a package.

Field Detail

hash

private static Hashtable hash

A hashtable containing for each type of data its hierarchy

Constructor Detail

MofHelper

public MofHelper()

Method Detail

getFeatures

public static Vector getFeatures(MofClass metaObject, AbstractNode parent, NameAdapter nameAdapter)

Retrieves all the features of a MofClass (i.e References and Attributes)

Parameters:
  metaObject - The MofClass whose features will be retrieved
  parent - The parent node of this MofClass

Returns:
  a Vector with the features as Nodes.

getFeatures

public static Vector getFeatures(Reference metaObject, AbstractNode parent, NameAdapter nameAdapter)

Retrieves all the features of a Mof Reference (i.e References and Attributes)

Parameters:
  metaObject - The Reference whose features will be retrieved.
  parent - The parent node of this class

Returns:
  a vector with the features as Nodes.
**getFeatures**

```java
public static Vector getFeatures(Attribute metaObject, AbstractNode parent, NameAdapter nameAdapter)
```

Retrieves all the features of a Mof Attribute (i.e References and Attributes). Attribute may be have as type a complex class whose features are retrieved.

**Parameters:**
- `metaObject` - The Reference whose features will be retrieved.
- `parent` - The parent node of this class

**Returns:**
- a vector with the features as Nodes or null if Attribute is of primitive type.

---

**getFeatures**

```java
public static Vector getFeatures(RefObject object, AbstractNode parent, NameAdapter nameAdapter)
```

Retrieves all the features of a RefObject (i.e M1 objects). Attributes may have as type a complex class whose features are retrieved.

**Parameters:**
- `object` - The RefObject whose features will be retrieved.
- `parent` - The parent node of this class

**Returns:**
- a vector with the features as Nodes or null if Attribute is of primitive type.

---

**getFeatures**

```java
public static Vector getFeatures(RefObject object)
```

Retrieves all the features of a RefObject (i.e M1 objects). Attributes may have as type a complex class whose features are retrieved.

**Parameters:**
- `object` - The RefObject whose features will be retrieved.

**Returns:**
- a vector with the features as Nodes or null if Attribute is of primitive type.

---

**newNode**

```java
private static AbstractNode newNode(RefObject object, AbstractNode parent, NameAdapter nameAdapter)
```

---

**getObject**

```java
private static RefObject getObject(RefObject object)
```
**Class MofHelper**

### getAllContents

```java
private static Vector getAllContents(MofClass metaObject)
```

Gets the features of a MofClass.

**Parameters:**
- `metaObject`: The MofClass whose features will be returned

**Returns:**
- a Vector of MofClasses with the features.

### getAllContentsAsNodes

```java
public static Vector getAllContentsAsNodes(MofClass metaObject, AbstractNode parent, NameAdapter nameAdapter)
```

Gets the features of a MofClass as Nodes

**Parameters:**
- `metaObject`: The MofClass whose features will be returned
- `parent`: The parent Node of the MofClass

**Returns:**
- a vector with the features as Nodes.

### changeToNodes

```java
public static Vector changeToNodes(Vector input, AbstractNode parent, NameAdapter nameAdapter)
```

Changes Mof classes into Nodes depending on the type of each class

**Parameters:**
- `input`: A Vector with the features found
- `parent`: The parent Node of the features

**Returns:**
- a vector with the features as Nodes.

### retrieveHierarchy

```java
private static void retrieveHierarchy(String metamodel)
```

Constructs a hash and retrieves in it the class hierarchies of SSL's Core, Instantiation and Types packages. These hierarchies are then used for retrieving the children of an abstract class.

### retrievePackageHierarchy

```java
private static void retrievePackageHierarchy(RefPackage pack)
```

Adds to a hash the class hierarchies of a package. These hierarchies are then used for retrieving the children of an abstract class.
Class MofHelper

Parameters:
  pack - The RefPackage whose class hierarchies will be retrieved.

Interface NameAdapter

org.dbc.kb.toolkit.proxyutils

All Known Implementing Classes:
  BmlInstanceNameAdapter, DefaultNameAdapter, ImmInstanceNameAdapter, OdmInstanceNameAdapter, SslInstanceNameAdapter

public interface NameAdapter

Author:
  gkoto

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Signature</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getId</td>
<td>String getId(RefBaseObject object)</td>
<td>73</td>
</tr>
<tr>
<td>getName</td>
<td>String getName(RefBaseObject object)</td>
<td>73</td>
</tr>
</tbody>
</table>

Method Detail

getName

public String getName(RefBaseObject object)

getId

public String getId(RefBaseObject object)

Class NavigationNode

org.dbc.kb.toolkit.proxyutils

java.lang.Object

org.dbc.kb.toolkit.proxyutils.AbstractNode

org.dbc.kb.toolkit.proxyutils.NavigationNode

public class NavigationNode
extends AbstractNode

The NavigationNode class contains a navigable object that has children. Navigable object can be all the model elements that have any type of feature; i.e Attributes, Operations, AssociationEnds etc. They are nodes of the tree control of FormulateQueryWizardPage class.

Author:
  George Kotopoulos
Class NavigationNode

Version:
1.0

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>children</td>
<td>Object[]</td>
<td>This node's children</td>
<td>74</td>
</tr>
<tr>
<td>object</td>
<td>RefObject</td>
<td>The Mof object of this NavigationNode.</td>
<td>74</td>
</tr>
</tbody>
</table>

Constructor Summary

Create a NavigationNode for a navigable object having a certain parent.

```
NavigationNode (RefObject node, AbstractNode parent, NameAdapter nameAdapter)
```

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>calcChildren()</td>
<td>void</td>
<td>Calculates the children of this node.</td>
<td>75</td>
</tr>
<tr>
<td>getChildren()</td>
<td>Object[]</td>
<td>Returns an array with this node's children.</td>
<td>76</td>
</tr>
<tr>
<td>getName()</td>
<td>String</td>
<td>Gets a String representation of the name of the node.</td>
<td>75</td>
</tr>
<tr>
<td>getObject()</td>
<td>Object</td>
<td>Gets the Object of the current node.</td>
<td>75</td>
</tr>
<tr>
<td>getParent()</td>
<td>AbstractNode</td>
<td>Gets the parent of the current node.</td>
<td>76</td>
</tr>
<tr>
<td>hasChildren()</td>
<td>boolean</td>
<td>Returns true if this node has children.</td>
<td>75</td>
</tr>
</tbody>
</table>

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode:
getChildren, getId, getName, getObject, getParent, hasChildren

Field Detail

**object**

```
private RefObject object
```

The Mof object of this NavigationNode. This object is navigable.

**children**

```
private Object[] children
```

This node's children
Constructor Detail

**NavigationNode**

```java
public NavigationNode(RefObject node,
                      AbstractNode parent,
                      NameAdapter nameAdapter)
```

Create a NavigationNode for a navigable object having a certain parent.

Method Detail

**calcChildren**

```java
private void calcChildren()
```

Calculates the children of this node. Children are computed on first demand and not when constructing the node for computational efficiency purposes.

**hasChildren**

```java
public boolean hasChildren()
```

Returns true if this node has children. False if it is a leaf node.

*Overrides:*

```java
hasChildren in class AbstractNode
```

*Returns:*

True if this node has children. False if it is a leaf node.

*See Also:*

```java
org.dbe.dtool.nodes.AbstractNode.hasChildren()
```

**getName**

```java
public String getName()
```

Gets a String representation of the name of the node.

*Overrides:*

```java
getName in class AbstractNode
```

*Returns:*

a String representation of the name of the node.

*See Also:*

```java
org.dbe.dtool.nodes.AbstractNode.getName()
```

**getObject**

```java
public Object getObject()
```
Class NavigationNode

Gets the Object of the current node. Each node is holding an object. It is a Mof ModelElement object.

**Overrides:**
- `getObject` in class `AbstractNode`

**Returns:**
- The Object of the current node

public `Object[] getChildren()`

Returns an array with this nodes children.

**Overrides:**
- `getChildren` in class `AbstractNode`

**Returns:**
- An array with this nodes children.

**See Also:**
- `org.dbe.dtool.nodes.AbstractNode.getChildren()`

getParent

public `AbstractNode getParent()`

Gets the parent of the current node.

**Overrides:**
- `getParent` in class `AbstractNode`

**Returns:**
- The parent of the current node.

**See Also:**
- `AbstractNode.getParent()`

Class OdmInstanceNameAdapter

```
org.dbe.kb.toolkit.proxyutils
java.lang.Object
    | org.dbe.kb.toolkit.proxyutils.OdmInstanceNameAdapter
```

All Implemented Interfaces:
- `NameAdapter`

public class `OdmInstanceNameAdapter`

extends `Object`
implements `NameAdapter`

<table>
<thead>
<tr>
<th>Constructor Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>OdmInstanceNameAdapter()</code></td>
<td>77</td>
</tr>
</tbody>
</table>

15/11/06
Class OdmInstanceNameAdapter

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Signature</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getId</td>
<td>String getId(RefBaseObject object)</td>
<td>77</td>
</tr>
<tr>
<td>getName</td>
<td>String getName(RefBaseObject object)</td>
<td>77</td>
</tr>
</tbody>
</table>

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
getId, getName

Constructor Detail

OdmInstanceNameAdapter

public OdmInstanceNameAdapter()

Method Detail

getName

public String getName(RefBaseObject object)

Specified by:
getName in interface NameAdapter

getId

public String getId(RefBaseObject object)

Specified by:
getId in interface NameAdapter

Class Proxy

org.dbe.kb.toolkit.proxyutils

java.lang.Object

org.dbe.kb.toolkit.proxyutils.Proxy

public class Proxy
extends Object

Author:
gkoto

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>bmlModelsTool</td>
<td>private org.dbe.kb.mdrman.JMtool</td>
<td>80</td>
</tr>
<tr>
<td>bmlTool</td>
<td>private org.dbe.kb.mdrman.JMtool</td>
<td>80</td>
</tr>
<tr>
<td>immModelsTool</td>
<td>private org.dbe.kb.mdrman.JMtool</td>
<td>80</td>
</tr>
</tbody>
</table>
### Constructor Summary

**Proxy**(Object service)

### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private void addModel(String metamodel, String id)</td>
<td>87</td>
</tr>
<tr>
<td>void clearTools()</td>
<td>88</td>
</tr>
<tr>
<td>void closeSession()</td>
<td>88</td>
</tr>
<tr>
<td>String getCurrentQuerySessionId()</td>
<td>89</td>
</tr>
<tr>
<td>private String getIMM()</td>
<td>82</td>
</tr>
<tr>
<td>private String getImmTool()</td>
<td>85</td>
</tr>
<tr>
<td>private void getModel(String metamodel, String id, boolean isSM, String nodeId)</td>
<td>87</td>
</tr>
<tr>
<td>void getModel(String metamodel, String id, boolean isSM)</td>
<td>86</td>
</tr>
<tr>
<td>RefPackage getModelPackage(String metamodel)</td>
<td>84</td>
</tr>
<tr>
<td>RefPackage getModelPackage(String metamodel, String nodeId, String modelId, boolean isSM)</td>
<td>84</td>
</tr>
<tr>
<td>String getNewSession()</td>
<td>88</td>
</tr>
<tr>
<td>private String getODM()</td>
<td>81</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool immTool</td>
<td>80</td>
</tr>
<tr>
<td>private org.dbe.kb.service.KBI kbService</td>
<td>79</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool odmModelsTool</td>
<td>81</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool odmTool</td>
<td>80</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool qmlTool</td>
<td>80</td>
</tr>
<tr>
<td>private String querySessionId</td>
<td>81</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool sdlModelsTool</td>
<td>80</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool sdlTool</td>
<td>80</td>
</tr>
<tr>
<td>private org.dbe.kb.sm.SMtool smTool</td>
<td>81</td>
</tr>
<tr>
<td>private org.dbe.kb.service.SRI srService</td>
<td>80</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool sslModelsTool</td>
<td>80</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool sslTool</td>
<td>80</td>
</tr>
<tr>
<td>Collection findDepSMs(String sessionId, String metamodel, String modelId)</td>
<td>87</td>
</tr>
<tr>
<td>void exportModel(ByteArrayOutputStream out)</td>
<td>83</td>
</tr>
<tr>
<td>void createQuerySession()</td>
<td>88</td>
</tr>
<tr>
<td>void closeSession(String sessionId)</td>
<td>88</td>
</tr>
<tr>
<td>void closeSession()</td>
<td>88</td>
</tr>
<tr>
<td>void clearTools()</td>
<td>88</td>
</tr>
<tr>
<td>private String getIMM()</td>
<td>82</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool getImmTool()</td>
<td>84</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool getImmTool(String modelId, String nodeId, String modelId, boolean isSM)</td>
<td>85</td>
</tr>
<tr>
<td>private org.dbe.kb.mdrman.JMItool getImmTool()</td>
<td>85</td>
</tr>
<tr>
<td>private String getODM()</td>
<td>81</td>
</tr>
<tr>
<td>private String getODM()</td>
<td>81</td>
</tr>
</tbody>
</table>
private org.dbe.kb.mdrman.JMItool getOdmTool() 85
private org.dbe.kb.mdrman.JMItool getOdmTool(String modelId) 85
private String getQML() 82
org.dbe.kb.metamodel.qml.QmlPackage getQMLpackage() 84
private org.dbe.kb.mdrman.JMItool getQmlTool() 85
Collection getResults() 83
Collection getResults(String sessionId) 83
private String getQML() 82
org.dbe.kb.metamodel.qml.QmlPackage getQMLpackage() 84
private org.dbe.kb.mdrman.JMItool getQmlTool() 85
String getSM(String smId, String nodeId) 83
org.dbe.kb.sm.SMtool getSMInfo(String smid, String nodeid) 88
private String getSDL() 81
String getSM() 82
private org.dbe.kb.mdrman.JMItool getSdlTool() 86
private org.dbe.kb.mdrman.JMItool getSdlTool(String modelId) 86
private org.dbe.kb.mdrman.JMItool getSdlTool(String modelId, String bmlModelId, String nodeId, boolean isSM) 86
private org.dbe.kb.mdrman.JMItool getSdlTool(String modelId, String nodeId, boolean isSM) 84
private org.dbe.kb.mdrman.JMItool getTool(String metamodel) 84
private org.dbe.kb.mdrman.JMItool getTool(String metamodel, String modelId, String nodeId, boolean isSM) 84
private void importModel(String metamodel, String modelId, String id, String nodeId, boolean isSM) 87
boolean isAlive() 88
boolean isSR() 86
void newQuery() 87
String retrieveBMLdata(String smId, String nodeId) 83
String retrieveBMLmodel(String modelId, String nodeId, boolean isSM) 83
String retrieveIMMmodel(String modelId) 82
void retrieveModel(String metamodel, String id, boolean isSM, String nodeId) 87
void retrieveModel(String metamodel, String id, String nodeId, boolean isSM) 87
String retrieveODMmodel(String modelId) 81
String retrieveSDLmodel(String modelId) 82
String retrieveSSLdata(String smId, String nodeId) 84
String retrieveSSLmodel(String modelId, String nodeId, boolean isSM) 82
Collection searchBMLByKeywords(String mm, Collection keywords) 88
Collection searchByKeywords(String mm, Collection keywords) 87
void setService(Object service) 81
Collection submitQuery() 83

Field Detail

kbService

private org.dbe.kb.service.KBI kbService
Class Proxy

srService
private org.dbe.kb.service.SRI srService

qmlTool
private org.dbe.kb.mdrman.JMItool qmlTool

sslTool
private org.dbe.kb.mdrman.JMItool sslTool

bmlTool
private org.dbe.kb.mdrman.JMItool bmlTool

sdlTool
private org.dbe.kb.mdrman.JMItool sdlTool

immTool
private org.dbe.kb.mdrman.JMItool immTool

odmTool
private org.dbe.kb.mdrman.JMItool odmTool

sslModelsTool
private org.dbe.kb.mdrman.JMItool sslModelsTool

bmlModelsTool
private org.dbe.kb.mdrman.JMItool bmlModelsTool

sdlModelsTool
private org.dbe.kb.mdrman.JMItool sdlModelsTool

immModelsTool
private org.dbe.kb.mdrman.JMItool immModelsTool
class Proxy

public class Proxy
{

    private org.dbe.kb.mdrman.JMItool odmModelsTool;

    private org.dbe.kb.sm.SMtool smTool;

    private String querySessionId;

    constructor detail

    public Proxy(Object service)
    {
    }

    method detail

    public void setService(Object service)
    {
    }

    private String getODM()
    {
        See Also:
        org.dbe.kb.service.KBI.getODM()
    }

    public String retrieveODMmodel(String modelId)
    {
        See Also:
        org.dbe.kb.service.KBI.retrieveODMmodel(java.lang.String)
    }

    private String getSSL()
    {
        See Also:
        org.dbe.kb.service.KBI.getSSL()
    }
}
retrieveSSLmodel

public String retrieveSSLmodel(String modelId,
    String nodeId,
    boolean isSM)

See Also:
    org.dbe.kb.service.KBI.retrieveSSLmodel(java.lang.String)

getSDL

private String getSDL()

See Also:
    org.dbe.kb.service.KBI.getSDL()

retrieveSDLmodel

public String retrieveSDLmodel(String modelId)

See Also:
    org.dbe.kb.service.KBI.retrieveSDLmodel(java.lang.String)

getBML

public String getBML()

See Also:
    org.dbe.kb.service.KBI.getBML()

getIMM

private String getIMM()

See Also:
    org.dbe.kb.service.KBI.getIMM()

retrieveIMMmodel

public String retrieveIMMmodel(String modelId)

See Also:
    org.dbe.kb.service.KBI.retrieveIMMmodel(java.lang.String)

getQML

private String getQML()
submitQuery

public Collection submitQuery() throws Exception

See Also:
org.dbe.kb.service.KBI.submitQuery(String, String)

getResults

public Collection getResults() throws Exception

See Also:
org.dbe.kb.service.KBI.getResults(String)

getResults

public Collection getResults(String sessionId)

See Also:
org.dbe.kb.service.KBI.getResults(String)

getSM

public String getSM(String smId, String nodeId)

See Also:
org.dbe.kb.service.SRI.getSM(String)

retrieveBMLdata

public String retrieveBMLdata(String smId, String nodeId)

See Also:
org.dbe.kb.service.SRI.retrieveBMLdata(String)

retrieveBMLmodel

public String retrieveBMLmodel(String modelId, String nodeId, boolean isSM)
retrieveSSLdata

public String retrieveSSLdata(String smId,
   String nodeId)

See Also:
org.dbe.kb.service.SRI.retrieveSSLdata(java.lang.String)

getQMLpackage

public org.dbe.kb.metamodel.qml.QmlPackage getQMLpackage()

getModelPackage

public RefPackage getModelPackage(String metamodel)

getModelPackage

public RefPackage getModelPackage(String metamodel,
   String nodeId,
   String modelId,
   boolean isSM)

getTool

private org.dbe.kb.mdrman.JMItool getTool(String metamodel,
   String modelId,
   String nodeId,
   boolean isSM)

getTool

private org.dbe.kb.mdrman.JMItool getTool(String metamodel)

getBmlTool

private org.dbe.kb.mdrman.JMItool getBmlTool()

Returns:
   Returns the bmlTool.
getBmlTool

private org.dbe.kb.mdrman.JMItool getBmlTool(String modelId,
        String bmlModelId,
        String nodeId,
        boolean isSM)

    Returns:
        Returns the bmlTool.

getImmTool

private org.dbe.kb.mdrman.JMItool getImmTool()

    Returns:
        Returns the immTool.

getImmTool

private org.dbe.kb.mdrman.JMItool getImmTool(String modelId)

    Returns:
        Returns the ImmTool.

getOdmTool

private org.dbe.kb.mdrman.JMItool getOdmTool()

    Returns:
        Returns the odmTool.

getOdmTool

private org.dbe.kb.mdrman.JMItool getOdmTool(String modelId)

    Returns:
        Returns the OdmTool.

getQmlTool

private org.dbe.kb.mdrman.JMItool getQmlTool()

    Returns:
        Returns the qmlTool.
**getSdlTool**

private org.dbe.kb.mdrman.JMItool getSdlTool()

**Returns:**
Returns the sdlTool.

**getSdlTool**

private org.dbe.kb.mdrman.JMItool getSdlTool(String modelId)

**Returns:**
Returns the SdlTool.

**getSslTool**

private org.dbe.kb.mdrman.JMItool getSslTool()

**Returns:**
Returns the sslTool.

**getSslTool**

private org.dbe.kb.mdrman.JMItool getSslTool(String modelId,
                                               String bmlModelId,
                                               String nodeId,
                                               boolean isSM)

**Returns:**
Returns the SslTool.

**isSR**

public boolean isSR()

**exportModel**

public void exportModel(ByteArrayOutputStream out)
                        throws IOException

**Throws:**
IOException

**getModel**

public void getModel(String metamodel,
                     String id,
                     String nodeId,
                     boolean isSM)
**getModel**

```java
public void getModel(String metamodel,
                      String id,
                      boolean isSM,
                      String nodeId)
```

**retrieveModel**

```java
public void retrieveModel(String metamodel,
                           String id,
                           String nodeId,
                           boolean isSM)
```

**retrieveModel**

```java
public void retrieveModel(String metamodel,
                           String id,
                           boolean isSM,
                           String nodeId)
```

**importModel**

```java
private void importModel(String metamodel,
                          String modelId,
                          String id,
                          String nodeId,
                          boolean isSM)
```

**addModel**

```java
private void addModel(String metamodel,
                       String id)
```

**findDepSMs**

```java
public Collection findDepSMs(String sessionId,
                              String metamodel,
                              String modelId)
```

**newQuery**

```java
public void newQuery()
```

**searchByKeywords**

```java
public Collection searchByKeywords(String mm,
                                    Collection keywords)
```
Class Proxy

searchBMLByKeywords

public Collection searchBMLByKeywords(String mm, 
Collection keywords)

getSMInfo

public org.dbe.kb.sm.SMtool getSMInfo(String smid, 
String nodeId)

closeSession

public void closeSession()

    Closes the current session

closeSession

public void closeSession(String sessionId)

createQuerySession

public void createQuerySession()

getNewSession

public String getNewSession()

getSSLModel

public String getSSLModel(String id, 
String nodeId)

getBMLModel

public String getBMLModel(String id, 
String nodeId)

isAlive

public boolean isAlive()

clearTools

public void clearTools()
getCurrentQuerySessionId

public String getCurrentQuerySessionId()

Class ProxyFinding

org.dbe.kb.toolkit.proxyutils

java.lang.Object
   ^org.dbe.kb.toolkit.proxyutils.ProxyFinding

public class ProxyFinding
extends Object

Field Summary

<table>
<thead>
<tr>
<th>private ProxyHandling proxy</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>89</td>
<td></td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>ProxyFinding()</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>89</td>
<td></td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>static String getContextPathForClass(String className, String metamodel)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static Vector getContextVectorItems(String metamodel)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>static Vector getContextVectorPaths(String metamodel)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ProxyHandling getProxyHandling()</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>89</td>
<td></td>
</tr>
</tbody>
</table>

Gets the shared instance of the ProxyHandler

Field Detail

proxy

private ProxyHandling proxy

Constructor Detail

ProxyFinding

public ProxyFinding()

Method Detail

getProxyHandling

public ProxyHandling getProxyHandling()

    Gets the shared instance of the ProxyHandler
The ProxyHandler class is responsible to connect to a FADA proxy and handle the whole connection and the communication. The Proxy can be set by an independent module, in order not to reconnect from this tool. To connect from an independent module see RegisterDiscoveryTool class.

Author:
George Kotopoulos

Version:
1.0

See Also:
org.dbe.studio.tools.discovery.RegisterDiscoveryTool

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static final String CONFIG_FILE</td>
<td>93</td>
</tr>
<tr>
<td>protected static Hashtable models</td>
<td>93</td>
</tr>
<tr>
<td>protected static Hashtable ontologies</td>
<td>93</td>
</tr>
<tr>
<td>private static Properties portalClientConfig</td>
<td>93</td>
</tr>
</tbody>
</table>
## Class ProxyHandling

### Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProxyHandling()</td>
<td>if no query proxy set.</td>
<td>93</td>
</tr>
</tbody>
</table>

### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>protected RefBaseObject changeEnvironment(RefBaseObject obj, String pathItem)</td>
<td></td>
</tr>
<tr>
<td>void closeSession()</td>
<td></td>
</tr>
<tr>
<td>void closeSession(String sessionId)</td>
<td></td>
</tr>
<tr>
<td>protected boolean containsOdmProperty(org.dbe.kb.metamodel.odm.properties.Property property, org.dbe.kb.metamodel.odm.classes.OdmClass odmClass)</td>
<td></td>
</tr>
<tr>
<td>void findProxy()</td>
<td>Connects to the FADA server of the properties (dtool.properties) file and finds a proxy.</td>
</tr>
<tr>
<td>Collection getAsynchronousResults()</td>
<td>Submits the query and returns a Vector with the results</td>
</tr>
<tr>
<td>static String getConfigString(String key)</td>
<td></td>
</tr>
<tr>
<td>RefBaseObject getContext(Vector path)</td>
<td>For a Vector of String constructs the context required.</td>
</tr>
<tr>
<td>String getCurrentQuerySessionId()</td>
<td></td>
</tr>
<tr>
<td>protected RefPackage getInstantiationModel()</td>
<td>Gets the SSL metamodel</td>
</tr>
<tr>
<td>RefPackage getMetamodel(String modelName)</td>
<td>Gets a metamodel's RefPackage</td>
</tr>
<tr>
<td>String getMetamodelName(int id)</td>
<td></td>
</tr>
<tr>
<td>Collection getModel(String metamodel, String modelId, String nodeId, boolean isSM)</td>
<td>For a string id of a model in a metamodel get the instance RefObject of the root</td>
</tr>
<tr>
<td>protected org.dbe.kb.metamodel.qml.contextdeclarations.QueryContextDecl getNewQuery(Vector path, String name, org.dbe.kb.metamodel.qml.ocl.expressions.OclExpression result)</td>
<td>Gets a new QueryContextDecl for a context path</td>
</tr>
<tr>
<td>String getNewSessionId()</td>
<td></td>
</tr>
<tr>
<td>Vector getObjects(String metamodel, String modelId, String nodeId, boolean isSM)</td>
<td>Returns all instances of the classes contained in this Package and its contained packages.</td>
</tr>
<tr>
<td>Vector getObjects(RefPackage refPack)</td>
<td>Returns all instances of the classes contained in this Package and its contained packages.</td>
</tr>
<tr>
<td>RefObject getOdmClass(String ontId, String id)</td>
<td>For a string id of a model in a metamodel get the instance RefObject of the root</td>
</tr>
</tbody>
</table>
Vector getOdmProperties(org.dbe.kb.metamodel.odm.classes.Odmclass odmclass)

For an OdmClass retrieve its properties

Vector getOdmRange(org.dbe.kb.metamodel.odm.properties.Objectl property)

For a string id of a model in a metamodel get the instar
RefObject of the root

private static Properties getPortalClientConfig()

getProxy()

Retrieves the current proxy object which has a number functions

org.dbe.kb.metamodel.qml.QmlPackage getQml()

Gets the current Oql Package.

Collection getResults(String sessionId)

RefClass getRootClass(String mm)

RefClass[] getRootClasses()

Vector getRoots(String mm)

protected RefClass getSemanticPackage()

Gets the SemanticPackage (the root class of the SSL m

Collection getServiceManifest(String sessionId, String metamodel, modelId)

org.dbe.kb.sm.SMtool getSMInfo(String smId, String nodeId)

protected RefPackage getSslModel()

Gets the SSL metamodel

RefPackage getSslSpecificPackage(String packName)

Gets an SSL's specific package ("Core", "Types", etc)

boolean isAlive()

boolean isInstantiatiable(String metamodel)

Return whether the metmodel specified by id is instant

boolean isSM(String smId)

boolean isSR()

Returns true if the proxy connected to is a ServiceRegi

protected org.dbe.kb.metamodel.odm.OdmPackage loadOntology(String id)

For a string id of a model in a metamodel get the instar
RefObject of the root

void newQuery()

Instantiates a new Query

void printOql()

Prints The QML Expression as an XMI in standard out

Object run()

Collection searchBMLByKeywords(String metamodel, Vector keywords)

Collection searchByKeywords(String metamodel, Vector keywords)

protected void setService(Object service)
Field Detail

proxy

private Proxy proxy

The Proxy

qml

protected org.dbe.kb.metamodel.qml.QmlPackage qml

The qml Package

ontologies

protected static Hashtable ontologies

models

protected static Hashtable models

portalClientConfig

private static Properties portalClientConfig

CONFIG_FILE

private static final String CONFIG_FILE

Constructor Detail

ProxyHandling

public ProxyHandling()

if no query proxy set. Connect to a FADA server and get a proxy.

Method Detail

getcxt

public RefBaseObject getcxt(Vector path)

For a Vector of String constructs the context reqiered. i.e A RefClass
Class ProxyHandling

getPortalClientConfig

private static Properties getPortalClientConfig()

getConfigString

public static String getConfigString(String key)
    throws IOException

    Throws:
    IOException

findProxy

public void findProxy()

    Connects to the FADA server of the properties (dtool.properties) file and finds a proxy.

setService

protected void setService(Object service)

newQuery

public void newQuery()

    Instantiates a new Query

getQml

public org.dbe.kb.metamodel.qml.QmlPackage getQml()

    Gets the current Oql Package. If none creates a new one.
    Returns:
    An Oql Package

getNewQuery

protected org.dbe.kb.metamodel.qml.contextdeclarations.QueryContextDecl getNewQuery(Vector path,
    String name,
    org.dbe.kb.metamodel.qml.ocl.expressions.OclExpression exp,
    String result)

    Gets a new QueryContextDecl for a context path
    Parameters:
    path - The context path (Vector of Strings)
name - The Query name
exp - The body ocl expression
result - The result type

**Returns:**
A new QueryContextDecl with the specific attributes.

---

**getMetamodel**

```java
public RefPackage getMetamodel(String modelName)
```

Gets a metamodel's RefPackage

**Parameters:**
- `modelName` - the metamodel's name (i.e. "SSL", "SDL", etc.)

**Returns:**
a metamodel's RefPackage

---

**getSslModel**

```java
protected RefPackage getSslModel()
```

Gets the SSL metamodel

**Returns:**
the SSL metamodel's RefPackage

---

**getInstantiationModel**

```java
protected RefPackage getInstantiationModel()
```

Gets the SSL metamodel

**Returns:**
the SSL metamodel's RefPackage

---

**getSemanticPackage**

```java
protected RefClass getSemanticPackage()
```

Gets the SemanticPackage (the root class of the SSL metamodel)

**Returns:**
the SemanticPackage's RefClass

---

**getRootClasses**

```java
public RefClass[] getRootClasses()
```

---

**getMetamodelName**

```java
public String getMetamodelName(int id)
```
getRootClass

public RefClass getRootClass(String mm)

Parameters:

mm - the metamodel's name. "SSL" or "SDL".

Returns:
The package's RefClass

Throws:

NotImplementedException - if mm is different from "SSL" or "SDL".

getSslSpecificPackage

public RefPackage getSslSpecificPackage(String packName)

Gets an SSL's specific package ("Core", "Types", etc)

Parameters:

packName - The SSL's package name ("Core", "Types", etc)

Returns:

A RefPackage object containing the required package

getAsynchronousResults

public Collection getAsynchronousResults()

Submits the query and returns a Vector with the results.

Returns:

A Vector with the results. Each result tuple contains the name of the result, a description and a rank.

Throws:

RuntimeException - if an error occurs on server or when submitting the query.

printOql

public void printOql()

Prints The QML Expression as an XMI in standard output.

isSR

public boolean isSR()

Returns true if the proxy connected to is a ServiceRegistry one. False otherwise

Returns:

True if the proxy connected to is a ServiceRegistry one. False otherwise
getModel

public Collection getModel(String metamodel,
                                  String id,
                                  String nodeId,
                                  boolean isSM)

For a string id of a model in a metamodel get the instance (the RefObject of the root)

Parameters:
  id - the id of the model required

Returns:
  the RefObject of the root of the model

changeEnvironment

protected RefBaseObject changeEnvironment(RefBaseObject obj,
                                      String pathItem)

loadOntology

protected org.dbe.kb.metamodel.odm.OdmPackage loadOntology(String id)

getOdmClass

public RefObject getOdmClass(String ontId,
                               String id)

getObjects

public Vector getObjects(RefPackage refPack)

Returns all instances of the classes contained in this Package and its contained packages.

Parameters:
  refPack - A RefPackage denoting the starting package.

Returns:
  A Vector of RefObject Objects
getObjects

public Vector getObjects(String metamodel, String modelId, String nodeId)

Returns all instances of the classes contained in this Package and its contained packages.
Returns:
A Vector of RefObject Objects

getOdmProperties

public Vector getOdmProperties(org.dbe.kb.metamodel.odm.classes.Odmclass odmclass)

For an OdmClass retrieve its properties
Parameters:
    odmclass - the OdmClass whose properties are retrieved
Returns:
A Vector with the properties of the odmclass

containsOdmProperty

protected boolean containsOdmProperty(org.dbe.kb.metamodel.odm.properties.Property property, org.dbe.kb.metamodel.odm.classes.Odmclass odmclass)

Returns:
true if the property exists in the class. false otherwise.

getOdmRange

public Vector getOdmRange(org.dbe.kb.metamodel.odm.properties.ObjectProperty property)

For a string id of a model in a metamodel get the instance (the RefObject of the root)
Parameters:
    property - the id of the model required
Returns:
the RefObject of the root of the model

isInstantiatable

public boolean isInstantiatable(String metamodel)

Return whether the metamodel specified by id is instantiatable
Parameters:
    metamodel - the metamodel name. Either "SSL" or "sdl"
Returns:
true if this metamodel is instanciatable. false otherwise.
**Class ProxyHandling**

**getNewSessionId**

public String `getNewSessionId()`

**getServiceManifest**

public Collection `getServiceManifest(String sessionId, String metamodel, String modelId)`

**Returns:**
a collection of Service Manifests

**run**

public Object `run()`
throws Exception

**Throws:**
Exception

**searchByKeywords**

public Collection `searchByKeywords(String metamodel, Vector keywords)`

**searchBMLByKeywords**

public Collection `searchBMLByKeywords(String metamodel, Vector keywords)`

**getSMInfo**

public org.dbe.kb.sm.SMtool `getSMInfo(String smId, String nodeId)`

**closeSession**

public void `closeSession()`

**closeSession**

public void `closeSession(String sessionId)`

**isSM**

public boolean `isSM(String smId)`
getProxy

public Proxy getProxy()

Retrieves the current proxy object which has a number of helper functions

Returns:
the current proxy object.

getRoots

public Vector getRoots(String mm)

isAlive

public boolean isAlive()

getResults

public Collection getResults(String sessionId)

getCurrentQuerySessionId

public String getCurrentQuerySessionId()

Class QueryNode

org.dbe.kb.toolkit.proxyutils

java.lang.Object

org.dbe.kb.toolkit.proxyutils.AbstractNode

org.dbe.kb.toolkit.proxyutils.QueryNode

All Implemented Interfaces:
IRootNode

public class QueryNode
extends AbstractNode
implements IRootNode

Author:
gkoto

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private int childIdx</td>
<td>102</td>
</tr>
<tr>
<td>private ContextRoot[] children</td>
<td>102</td>
</tr>
<tr>
<td>private string name</td>
<td>102</td>
</tr>
</tbody>
</table>
Class QueryNode

Fields inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
ALL, HARD, NONE, SOFT

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueryNode(String name)</td>
<td>102</td>
</tr>
<tr>
<td>QueryNode(Collection roots, String name, String metamodel)</td>
<td>102</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object[] getChildren()</td>
<td>Returns an array with this nodes children.</td>
</tr>
<tr>
<td>String getMetamodel()</td>
<td>Gets a String representation of the query name.</td>
</tr>
<tr>
<td>String getName()</td>
<td>Gets a String representation of the name of the node.</td>
</tr>
<tr>
<td>Iterator getNodes()</td>
<td>Gets an iterator with the ValueNodes of this query.</td>
</tr>
<tr>
<td>Iterator getNodes(boolean isHard)</td>
<td>Gets an Iterator with the ValueNodes of a certain type (hard or soft constraints)</td>
</tr>
<tr>
<td>int getNodesFlag()</td>
<td>Gets a int flag for the type of the ValueNodes.</td>
</tr>
<tr>
<td>Object getObject()</td>
<td>Gets the Object of the current node.</td>
</tr>
<tr>
<td>Vector getPath()</td>
<td>Gets a Vector containg the package path to the context Mof class of this query.</td>
</tr>
<tr>
<td>String getQueryName()</td>
<td>Gets a String representation of the query name.</td>
</tr>
<tr>
<td>String getResultType()</td>
<td>Gets the result type of this context, i.e.</td>
</tr>
<tr>
<td>ContextRoot getSelectedChild()</td>
<td>Returns true if this node has children.</td>
</tr>
<tr>
<td>boolean hasChildren()</td>
<td>Returns true if this node has children.</td>
</tr>
<tr>
<td>boolean hasNodes()</td>
<td>Returns true is the query has at least one value node, false otherwise.</td>
</tr>
<tr>
<td>boolean isInstance()</td>
<td>Always returns false</td>
</tr>
<tr>
<td>void setSelectedMetamodel(int idx)</td>
<td>105</td>
</tr>
<tr>
<td>void setSelectedMetamodel(ContextRoot node)</td>
<td>105</td>
</tr>
</tbody>
</table>

Methods inherited from class org.dbe.kb.toolkit.proxyutils.AbstractNode
getChildren, getId, getName, getObject, getParent, hasChildren

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.IRootNode
getChildren, getMetamodel, getNodes, getNodes, getNodesFlag, getPath, getQueryName, getResultType, hasNodes, isInstance
Field Detail

name
private String name

children
private ContextRoot[] children

childIdx
private int childIdx

Constructor Detail

QueryNode
public QueryNode(String name)

QueryNode
public QueryNode(Collection roots, 
        String name, 
        String metamodel)

Method Detail

hasChildren
public boolean hasChildren()

    Description copied from class: AbstractNode
    Returns true if this node has children. False if it is a leaf node.
    Overrides: 
        hasChildren in class AbstractNode
    Returns: 
        True if this node has children. False if it is a leaf node.
    See Also: 
        org.dbe.dtool.nodes.AbstractNode.hasChildren()

getChildren
public Object[] getChildren()

    Description copied from class: AbstractNode
    Returns an array with this nodes children.
    Specified by: 
        getChildren in interface IRootNode
Overrides:

getChildren in class AbstractNode

Returns:
An array with this nodes children.

See Also:
org.dbe.dtool.nodes.AbstractNode.getChildren()

---

**getObject**

public Object getObject()

Description copied from class: AbstractNode

Gets the Object of the current node. Each node is holding an object. Usually a MOF object

Overrides:

getObject in class AbstractNode

Returns:
The Object of the current node

See Also:
org.dbe.dtool.nodes.AbstractNode.getObject()

---

**getName**

public String getName()

Description copied from class: AbstractNode

Gets a String representation of the name of the node.

Overrides:

getName in class AbstractNode

Returns:
a String representation of the name of the node.

See Also:
org.dbe.dtool.nodes.AbstractNode.getName()

---

**getNodesFlag**

public int getNodesFlag()

Description copied from interface: IRootNode

Gets a int flag for the type of the ValueNodes. Valid values are NONE (0), HARD (1), SOFT (2) and ALL (3)

Specified by:

getRootNode in interface IRootNode

Returns:
NONE (0), HARD (1), SOFT (2) and ALL (3).

See Also:
org.dbe.dtool.nodes.IRootNode.getNodesFlag()
**Class QueryNode**

**getNodes**

public Iterator getNodes(boolean isHard)

*Description copied from interface: IRootNode*

Gets an Iterator with the ValueNodes of a certain type (hard or soft constrains)

*Specified by:*

getNodes in interface IRootNode

*Parameters:*

isHard - if the ValueNodes to be returned are hard constrains or not.

*Returns:*

An Iterator with the ValueNodes of a certain type (hard or soft constrains)

*See Also:*

org.dbe.dtool.nodes.IRootNode.getNodes(boolean)

---

**getQueryName**

public String getQueryName()

*Description copied from interface: IRootNode*

Gets a String representation of the query name.

*Specified by:*

getQueryName in interface IRootNode

*Returns:*

a String representation of the query name.

*See Also:*

org.dbe.dtool.nodes.IRootNode.getQueryName()

---

**hasNodes**

public boolean hasNodes()

*Description copied from interface: IRootNode*

Returns true is the query has at least one value node, false otherwise.

*Specified by:*

hasNodes in interface IRootNode

*Returns:*

ture is the query has at least one value node, false otherwise.

*See Also:*

org.dbe.dtool.nodes.IRootNode.hasNodes()

---

**getNodes**

public Iterator getNodes()

*Description copied from interface: IRootNode*

Gets an iterator with the ValueNodes of this query.

*Specified by:*

getNodes in interface IRootNode
**Class QueryNode**

**Returns:**
The ValueNodes of this query.

**See Also:**
org.dbe.dtool.nodes.IRootNode.getNodes()

---

**isInstance**

```java
public boolean isInstance()
```

Always returns false

**Specified by:**
isInstance in interface IRootNode

**Returns:**
false

**See Also:**
org.dbe.dtool.nodes.IRootNode.isInstance()

---

**setSelectedMetamodel**

```java
public void setSelectedMetamodel(int idx)
```

---

**setSelectedMetamodel**

```java
public void setSelectedMetamodel(ContextRoot node)
```

---

**getSelectedChild**

```java
public ContextRoot getSelectedChild()
```

---

**getMetamodel**

```java
public String getMetamodel()
```

**Description copied from interface: IRootNode**

Gets a String representation of the query name.

**Specified by:**
getMetamodel in interface IRootNode

**Returns:**
a String representation of the query name.

---

**getPath**

```java
public Vector getPath()
```

**Description copied from interface: IRootNode**

Gets a Vector containting the package path to the context Mof class of this query.

**Specified by:**
getPath in interface IRootNode
Class QueryNode

Returns:
   a Vector containg the package path to the context M of class of this query.

getResultType

public String getResultType()

Description copied from interface: IRootNode
   Gets the result type of this context, i.e. the metamodel.
   Specified by:
      getResultType in interface IRootNode
   Returns:
      The result type of this context, i.e. the metamodel.

Class SslInstanceNameAdapter

org.dbe.kb.toolkit.proxyutils

java.lang.Object
   org.dbe.kb.toolkit.proxyutils.SslInstanceNameAdapter

All Implemented Interfaces:
   NameAdapter

public class SslInstanceNameAdapter
   extends Object
   implements NameAdapter

Author:
   gkoto

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SslInstanceNameAdapter()</td>
<td>106</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getId(RefBaseObject object)</td>
<td>107</td>
</tr>
<tr>
<td>String getName(RefBaseObject object)</td>
<td>107</td>
</tr>
</tbody>
</table>

Methods inherited from interface org.dbe.kb.toolkit.proxyutils.NameAdapter
   getId, getName

Constructor Detail

SslInstanceNameAdapter

public SslInstanceNameAdapter()
Method Detail

getName

public String getName(RefBaseObject object)

Specified by:
getName in interface NameAdapter

getId

public String getId(RefBaseObject object)

Specified by:
getId in interface NameAdapter

Class ValueNode

org.dbe.kb.toolkit.proxyutils
java.lang.Object

public class ValueNode
extends AbstractNode

The ValueNode class represents a constraint (either hard or soft) on it's parent AttributeNode. They are nodes of the tree control of FormulateQueryWizardPage class.

Author:
George Kotopoulos

Version:
1.0

Field Summary

<table>
<thead>
<tr>
<th>static Object[] EMPTY_ARRAY</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private boolean isHard</td>
<td>109</td>
</tr>
<tr>
<td>private String operation</td>
<td>108</td>
</tr>
<tr>
<td>private String pathString</td>
<td>109</td>
</tr>
<tr>
<td>private String value</td>
<td>108</td>
</tr>
<tr>
<td>private int weight</td>
<td>109</td>
</tr>
</tbody>
</table>
Class ValueNode

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ValueNode(String op, String value, int weight, AbstractNode parent)</td>
<td>109</td>
</tr>
<tr>
<td>Creates a new ValueNode with the specific values</td>
<td></td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object[] getChildren()</td>
<td>109</td>
</tr>
<tr>
<td>Returns an array with this nodes children.</td>
<td></td>
</tr>
<tr>
<td>String getName()</td>
<td>110</td>
</tr>
<tr>
<td>Gets a String representation of the name of the node.</td>
<td></td>
</tr>
<tr>
<td>Object getObjectName()</td>
<td>110</td>
</tr>
<tr>
<td>Gets the parent object of this node.</td>
<td></td>
</tr>
<tr>
<td>String getOperation()</td>
<td>111</td>
</tr>
<tr>
<td>Gets the selected operation</td>
<td></td>
</tr>
<tr>
<td>AbstractNode getParent()</td>
<td>110</td>
</tr>
<tr>
<td>Gets the parent of the current node.</td>
<td></td>
</tr>
<tr>
<td>String getValue()</td>
<td>111</td>
</tr>
<tr>
<td>The assigned value</td>
<td></td>
</tr>
<tr>
<td>int getWeight()</td>
<td>111</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>boolean hasChildren()</td>
<td>109</td>
</tr>
<tr>
<td>Returns true if this node has children.</td>
<td></td>
</tr>
<tr>
<td>boolean isHard()</td>
<td>111</td>
</tr>
<tr>
<td>Flag suggesting if it is a hard constraint (true) or not</td>
<td></td>
</tr>
<tr>
<td>void setValue(String op, String val, int weight)</td>
<td>110</td>
</tr>
<tr>
<td>Sets the value of this node.</td>
<td></td>
</tr>
<tr>
<td>String toString()</td>
<td>111</td>
</tr>
<tr>
<td>Returns a string representation.</td>
<td></td>
</tr>
</tbody>
</table>

Methods inherited from class org.dbe.kb.toolkit.proxutils.AbstractNode

getChildren, getId, getName, getObject, getParent, hasChildren

Field Detail

operation

private String operation

a String representation of the operation of this constraint

value

private String value

The literal value that constraints the Attribute
### Class ValueNode

**pathString**

private String `pathString`

A string representation of the path to the tree root.

---

**isHard**

private boolean `isHard`

True if this constraint is a hard one, false otherwise.

---

**EMPTY_ARRAY**

public static Object[] `EMPTY_ARRAY`

---

**weight**

private int `weight`

---

#### Constructor Detail

**ValueNode**

public `ValueNode`(String `op`,
String `value`,
int `weight`,
`AbstractNode` `parent`)

Creates a new ValueNode with the specific values

---

#### Method Detail

**hasChildren**

public boolean `hasChildren()`

Returns true if this node has children. False if it is a leaf node.

**Overrides:**

`hasChildren` in class `AbstractNode`

**Returns:**

True if this node has children. False if it is a leaf node.

**See Also:**

`org.dbe.dtool.nodes.AbstractNode.hasChildren()`

---

**getChildren**

public Object[] `getChildren()`
Class ValueNode

Returns an array with this nodes children.

**Overrides:**
```
getChildren in class AbstractNode
```

**Returns:**
An array with this nodes children.

**See Also:**
```
org.dbe.dtool.nodes.AbstractNode.getChildren()
```

---

**getName**

```java
public String getName()
```

Gets a String representation of the name of the node.

**Overrides:**
```
getName in class AbstractNode
```

**Returns:**
a String representation of the name of the node.

**See Also:**
```
org.dbe.dtool.nodes.AbstractNode.getName()
```

---

**getObject**

```java
public Object getObject()
```

Gets the parent object of this node. Parent is usually of type AttributeNode. So, An Mof Attribute is returned.

**Overrides:**
```
getObject in class AbstractNode
```

**Returns:**
The Object of the current node

**See Also:**
```
org.dbe.dtool.nodes.AbstractNode.getObject()
```

---

**getParent**

```java
public AbstractNode getParent()
```

Gets the parent of the current node.

**Overrides:**
```
getParent in class AbstractNode
```

**Returns:**
The parent of the current node.

**See Also:**
```
AbstractNode.getParent()
```

---

**setValue**

```java
public void setValue(String op,
                     String val,
                     int weight)
```

---

15/11/06
Sets the value of this node.

**Parameters:**
- `op` - The operation selected
- `val` - the value assigned
- `weight` - a flag suggesting whether it is a hard or a soft constraint

---

getOperation

```java
public String getOperation()
```

Gets the selected operation

**Returns:**
the operation

---

getValue

```java
public String getValue()
```

The assigned value

**Returns:**
the assigned value

---

getWeight

```java
public int getWeight()
```

---

isHard

```java
public boolean isHard()
```

Flag suggesting if it is a hard constraint (true) or not

**Returns:**
true if it a Hard constraint, false otherwise.

---

toString

```java
public String toString()
```

Returns a string representation of this constraint.

**Overrides:**
`toString in class Object`

---

**Package org.dbe.toolkit.portal**
public class DBEPortal
extends Object

Author:
andy-edmonds

Field Summary

Field Detail

helper
private org.sun.dbe.ClientHelper helper

Constructor Detail

DBEPortal
public DBEPortal(org.sun.dbe.ClientHelper helper)
Method Detail

execute

public Object[] execute(String smid,
                        String methodName,
                        Object[] parameters)
    throws DBEPortalException

Parameters:
  smid - the SMID of the service to execute
  methodName - the name of the method to invoke
  parameters - the list of parameters to supply to the method invocation

Returns:
  returns a list containing the results; 1st element is the return value of ServiceProxy.invoke

Throws:
  DBEPortalException

See Also:
  IDBEPortal

getUI

public String getUI(String smid)
    throws DBEPortalException

Parameters:
  smid - the SMID of the service that will supply the service UI

Returns:
  a URL represented as a String pointing to the UI that the client can use

Throws:
  DBEPortalException

See Also:
  IDBEPortal

eexecuteIF

public Object executeIF(String smid,
                         HashMap params)
    throws DBEPortalException

Parameters:
  smid - the SMID of the service that will supply the service UI

Returns:
  a URL represented as a String pointing to the UI that the client can use

Throws:
  DBEPortalException

See Also:
  IDBEPortal
### Class DBEPortalException

```java
org.dbe.toolkit.portal
```

java.lang.Object
  java.lang.Throwable
    java.lang.Exception
      org.dbe.toolkit.portal.DBEPortalException

All Implemented Interfaces:
  Serializable

public class DBEPortalException
extends Exception

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static final long serialVersionUID</td>
<td>114</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Constructor Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBEPortalException(String string)</td>
<td>114</td>
</tr>
</tbody>
</table>

### Field Detail

serialVersionUID

private static final long serialVersionUID

### Constructor Detail

DBEPortalException

public DBEPortalException(String string)

### Class DBEPortalToolkitConfig

```java
org.dbe.toolkit.portal
```

java.lang.Object
  org.dbe.toolkit.portal.DBEPortalToolkitConfig

public class DBEPortalToolkitConfig
extends Object
Field Summary

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static final String CONFIG_FILE</td>
<td>115</td>
</tr>
<tr>
<td>private static Properties portalServiceConfig</td>
<td>115</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private DBEPortalToolkitConfig()</td>
<td>115</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static String getString(String key) throws IOException</td>
<td>115</td>
</tr>
</tbody>
</table>

Field Detail

**portalServiceConfig**

private static Properties portalServiceConfig

**CONFIG_FILE**

private static final String CONFIG_FILE

Constructor Detail

**DBEPortalToolkitConfig**

private DBEPortalToolkitConfig()

Default constructor

Method Detail

**getString**

public static String getString(String key) throws IOException

Gets a configuration parameter as a String

Parameters:

key - the requested parameter name

Returns:

the value of the parameter name

Throws:

IOException
getPortalServiceConfig

private static Properties getPortalServiceConfig()

Gets the portal configuration Properties bundle

Returns:
the properties bundle

Package org.dbe.toolkit.portal.client

Class Summary

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBEPortalClient</td>
<td>116</td>
</tr>
<tr>
<td>ProxyCache</td>
<td>120</td>
</tr>
<tr>
<td>ProxyObject</td>
<td>123</td>
</tr>
</tbody>
</table>

Class DBEPortalClient

org.dbe.toolkit.portal.client

java.lang.Object

Lorg.dbe.toolkit.portal.client.DBEPortalClient

public class DBEPortalClient extends Object

Author:
andy-edmonds

Field Summary

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static final String CONFIG_FILE</td>
<td>117</td>
</tr>
<tr>
<td>private static org.sun.dbe.ClientHelper helper</td>
<td>117</td>
</tr>
<tr>
<td>private static final String HTTP</td>
<td>117</td>
</tr>
<tr>
<td>private static org.apache.log4j.Logger logger</td>
<td>117</td>
</tr>
<tr>
<td>private static final String PORT_SEP</td>
<td>117</td>
</tr>
<tr>
<td>private static Properties portalClientConfig</td>
<td>118</td>
</tr>
<tr>
<td>private static ProxyCache proxyMan</td>
<td>118</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBEPortalClient()</td>
<td>118</td>
</tr>
</tbody>
</table>

Default constructor
### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>public Object[] executeIF(String smId, String smIdUser, String inObj)</td>
<td>120</td>
</tr>
<tr>
<td>static String getConfigString(String key)</td>
<td>119</td>
</tr>
<tr>
<td>static org.apache.log4j.Logger getLogger()</td>
<td>119</td>
</tr>
<tr>
<td>private static Properties getPortalClientConfig()</td>
<td>119</td>
</tr>
<tr>
<td>static ProxyCache getProxyManager()</td>
<td>119</td>
</tr>
<tr>
<td>static String getUI(String smid)</td>
<td>118</td>
</tr>
<tr>
<td>void init()</td>
<td>118</td>
</tr>
<tr>
<td>List invoke(String smid, String methodName, Vector parameters)</td>
<td>118</td>
</tr>
<tr>
<td>boolean isAlive(String smid)</td>
<td>119</td>
</tr>
<tr>
<td>private void setLoggingLevel()</td>
<td>118</td>
</tr>
</tbody>
</table>

### Field Detail

**PORT_SEP**

private static final String PORT_SEP

**HTTP**

private static final String HTTP

**CONFIG_FILE**

private static final String CONFIG_FILE

**helper**

private static org.sun.dbe.ClientHelper helper

**logger**

private static org.apache.log4j.Logger logger
**Class DBEPortalClient**

**portalClientConfig**

private static Properties portalClientConfig

**proxyMan**

private static ProxyCache proxyMan

---

**Constructor Detail**

**DBEPortalClient**

public DBEPortalClient()

Default constructor

---

**Method Detail**

**init**

public void init()

Initialises the portal client and enumerates it's configuration parameters

---

**setLoggingLevel**

private void setLoggingLevel()

Sets the logging level to be used by the Portal client

---

**invoke**

public List invoke(String smid,
                   String methodName,
                   Vector parameters)

Invokes a DBE service

**Parameters:**

- smid - SMID of the target service to invoke
- methodName - method of the service to invoke
- parameters - a list of parameter values, ordered as the method signature expects them

**Returns:**

an ordered list of return results, the first entry in this list is always the return value of Workspace.invoke()

---

**getUI**

public String getUI(String smid)
Class DBEPortalClient

Returns the URL of where the requested UI or IF UI is located on the servent

Parameters:

- **smid** - The SMID of the service which has the requested UI or IF UI

Returns:
The URL of the UI or IF UI

getLogger

culbrt static org.apache.log4j.Logger getLogger()

cgetConfigString

culbrt static String getConfigString(String key)

Throws:

- IOException

getProxyManager

culbrt static ProxyCache getProxyManager()

Throws:

- DBEPortalException

getPortalClientConfig

private static Properties getPortalClientConfig()

Returns:
a properties bundle containing the configuration

isAlive

public boolean isAlive(String smid)

Throws:

- DBEPortalException
executeIF

```java
public Object[] executeIF(String smId,
            String smIdUser,
            String inObj)
```

**Parameters:**
- `smId` - The Service manifest ID
- `smIdUser` - The Service manifest ID of the user service
- `inObj` - The data (xml string) that made the service manifest

**Returns:**
the SM

**Author:**
mbordin

---

## Class ProxyCache

**org.dbe.toolkit.portal.client**

**java.lang.Object**

L **org.dbe.toolkit.proxyframework.Workspace**

**public class ProxyCache**

extends Object

---

### Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cacheEnabled</code></td>
<td><code>private boolean</code></td>
<td>121</td>
</tr>
<tr>
<td><code>helper</code></td>
<td><code>private org.sun.dbe.ClientHelper</code></td>
<td>121</td>
</tr>
<tr>
<td><code>logger</code></td>
<td><code>private static org.apache.log4j.Logger</code></td>
<td>121</td>
</tr>
<tr>
<td><code>OBJ_TTL</code></td>
<td><code>private static long</code></td>
<td>121</td>
</tr>
<tr>
<td><code>proxyCache</code></td>
<td><code>private static HashMap</code></td>
<td>121</td>
</tr>
</tbody>
</table>

### Constructor Summary

**ProxyCache**(org.sun.dbe.ClientHelper helper)

**Constructor**

121

### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>get(String smid)</code></td>
<td><code>org.dbe.toolkit.proxyframework.Workspace</code></td>
<td>122</td>
</tr>
<tr>
<td><code>getCache()</code></td>
<td><code>private HashMap</code></td>
<td>122</td>
</tr>
<tr>
<td><code>getWorkspace(String smid, org.dbe.toolkit.proxyframework.Workspace ws)</code></td>
<td><code>org.dbe.toolkit.proxyframework.Workspace</code></td>
<td>122</td>
</tr>
<tr>
<td><code>isCached(String smid)</code></td>
<td><code>private boolean</code></td>
<td>123</td>
</tr>
<tr>
<td><code>isCacheEnabled()</code></td>
<td><code>boolean</code></td>
<td>121</td>
</tr>
</tbody>
</table>
Class ProxyCache

```java
private ProxyObject retrieve(String smid)

Retrieves the proxy from the cache corresponding to the supplied SMID
```

```java
void setCacheEnabled(boolean cacheEnabled)

Check to see if proxy caching is enabled or not
```

```java
private void store(String smid, org.dbe.toolkit.proxyframework.Workspace ws)

Stores the proxy to the cache
```

Field Detail

**helper**

private org.sun.dbe.ClientHelper helper

**cacheEnabled**

private boolean cacheEnabled

**logger**

private static org.apache.log4j.Logger logger

**proxyCache**

private static HashMap proxyCache

**OBJ_TTL**

private static long OBJ_TTL

Constructor Detail

**ProxyCache**

public ProxyCache(org.sun.dbe.ClientHelper helper)

Constructor

Method Detail

**isCacheEnabled**

public boolean isCacheEnabled()
Class ProxyCache

setCacheEnabled

public void setCacheEnabled(boolean cacheEnabled)

    Check to see if proxy caching is enabled or not

getCache

private HashMap getCache()

    gets the cache of proxies

get

public org.dbe.toolkit.proxyframework.Workspace get(String smid)
    throws DBEPortalException

    Gets a proxy of the corresponding SMID
    Parameters:
        smid - the requested service
    Returns:
        proxy of the requested service
    Throws:
        DBEPortalException

getWorkspace

private org.dbe.toolkit.proxyframework.Workspace getWorkspace(String smid,
        org.dbe.toolkit.proxyframework.Workspace ws)
    throws DBEPortalException

    Gets the workspace of the requested service (SMID)
    Throws:
        DBEPortalException

store

private void store(String smid,
        org.dbe.toolkit.proxyframework.Workspace ws)

    Stores the proxy to the cache
    Parameters:
        smid - SMID of the proxy
        ws - the actual proxy of the service

retrive

private ProxyObject retrive(String smid)
Class ProxyCache

Retrieves the proxy from the cache corresponding to the supplied SMID

Parameters:
  smid - SMID of the proxy to be retrieved

Returns:
  the proxy

isCached

private boolean isCached(String smid)

Check to see if the proxy is cached locally

Class ProxyObject

org.dbe.toolkit.portal.client

java.lang.Object
  org.dbe.toolkit.portal.client.ProxyObject

public class ProxyObject
extends Object

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>proxy</td>
<td>org.dbe.toolkit.proxyframework.Workspace</td>
<td></td>
</tr>
<tr>
<td>timestamp</td>
<td>long</td>
<td></td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProxyObject</td>
<td>(org.dbe.toolkit.proxyframework.Workspace ws, long timestamp)</td>
<td></td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getProxy</td>
<td>()</td>
<td>124</td>
</tr>
<tr>
<td>getTimestamp</td>
<td>()</td>
<td>124</td>
</tr>
</tbody>
</table>

Field Detail

proxy

org.dbe.toolkit.proxyframework.Workspace proxy

timestamp

long timestamp
Class ProxyObject

Constructor Detail

ProxyObject

public ProxyObject(org.dbe.toolkit.proxyframework.Workspace ws, long timestamp)

Method Detail

getProxy

public org.dbe.toolkit.proxyframework.Workspace getProxy()

getTimestamp

public long getTimestamp()

Package org.dbe.toolkit.portal.execution

Class Summary

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExecutionDelegate</td>
<td>124</td>
</tr>
</tbody>
</table>

Class ExecutionDelegate

org.dbe.toolkit.portal.execution

java.lang.Object

org.dbe.toolkit.portal.execution.ExecutionDelegate

public class ExecutionDelegate
extends Object

Author:
andy-edmonds

Field Summary

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static final String BOOLEAN</td>
<td>126</td>
</tr>
<tr>
<td>private static final String DATE_TIME</td>
<td>125</td>
</tr>
<tr>
<td>private static final String EXECUTE_IF</td>
<td>126</td>
</tr>
<tr>
<td>private org.sun.dbe.ClientHelper helper</td>
<td>126</td>
</tr>
<tr>
<td>private Vector holderObjects</td>
<td>126</td>
</tr>
<tr>
<td>private static final String INTEGER</td>
<td>126</td>
</tr>
<tr>
<td>private static final String log4j.Logger logger</td>
<td>127</td>
</tr>
<tr>
<td>private Vector outParamHolders</td>
<td>126</td>
</tr>
<tr>
<td>private Class[] outParamHoldersArray</td>
<td>126</td>
</tr>
</tbody>
</table>
private static final String REAL

private String smid

private static final String STRING

private static final String TMP_SDL_POSTFIX

private static final String TMP_SDL_PREFIX

private static final String URI

### Constructor Summary

**ExecutionDelegate**(org.sun.dbe.ClientHelper helper, String smid)

### Method Summary

**dumpSDL**(String sdlStr)
- Takes a SDL file and dumps it to string

**execute**(String methodName, Object[] parameters)
- Executes the method with supplied parameters

**executeIF**(HashMap parameters)
- Gets the value of a holder object based on type introspection

**getInClass**(org.dbe.studio.editor.sdl.Type type)
- Finds the java

**getInvocationResults**(Object ret, Object[] parameters)
- Merges out parameters and result of Workspace.invoke() into one array

**getMethodParametersTypes**(String methodName)
- Gets the parameter types for a particular method

**getMethodParametersValues**(Object[] parameters)
- Gets the

**getOutClass**(org.dbe.studio.editor.sdl.Type type, int j)
- Gets the

**getWorkspace**()
- Gets a Workspace object

### Field Detail

**URI**

private static final String URI

**DATE_TIME**

private static final String DATE_TIME
REAL
private static final String REAL

INTEGER
private static final String INTEGER

BOOLEAN
private static final String BOOLEAN

STRING
private static final String STRING

TMP_SDL_POSTFIX
private static final String TMP_SDL_POSTFIX

TMP_SDL_PREFIX
private static final String TMP_SDL_PREFIX

EXECUTE_IF
private static final String EXECUTE_IF

helper
private org.sun.dbe.ClientHelper helper

outParamHolders
private Vector outParamHolders

outParamHoldersArray
private Class[] outParamHoldersArray

holderObjects
private Vector holderObjects
Class ExecutionDelegate

smid

private String smid

logger

private static org.apache.log4j.Logger logger

Constructor Detail

ExecutionDelegate

public ExecutionDelegate(org.sun.dbe.ClientHelper helper,
       String smid)

Method Detail

executeIF

public Object executeIF(HashMap parameters)
   throws DBEPortalException

   Throws:
       DBEPortalException

execute

public Object[] execute(String methodName,
               Object[] parameters)
   throws DBEPortalException

   Executes the method with supplied parameters
   Parameters:
       methodName - Name of the method to execute
       parameters - parameters to supply to the the method
   Returns:
       Results of the exection
   Throws:
       DBEPortalException

getWorkspace

private org.dbe.toolkit.proxyframework.Workspace getWorkspace()
   throws DBEPortalException

   Gets a Workspace object
   Returns:
       The workspace object
   Throws:
       DBEPortalException
getMethodParametersTypes

private Class[] getMethodParametersTypes(String methodName) throws DBEPortalException

Gets the parameter types for a particular method

Parameters:
- methodName - The method to get the parameters types

Returns:
- A list of method parameter types in the order of the method’s signature

Throws:
- DBEPortalException

dumpSDL

private String dumpSDL(String sdlStr)

Takes a SDL file and dumps it to string

Parameters:
- sdlStr - the absolute path of a SDL file

Returns:
- The SDL as a string

getInClass

private Class getInClass(org.dbe.studio.editor.sdl.Type type) throws DBEPortalException

Finds the java

Parameters:
- type - type to get Class object of

Returns:
- Class object supplied

Throws:
- DBEPortalException

See Also:
- object of the type supplied

getOutClass

private Class getOutClass(org.dbe.studio.editor.sdl.Type type, int j) throws DBEPortalException

Gets the

Parameters:
- type - Type to find the
- j - a reference counter
Class ExecutionDelegate

**Returns:**
Class object of the supplied type

**Throws:**
*DBEPortalException*
*Exception*

**See Also:**
object of the supplied type, object of

ggetMethodParametersValues

```java
private Object[] getMethodParametersValues(Object[] parameters)
throws DBEPortalException
```

**Throws:**
*DBEPortalException*

ggetInvocationResults

```java
private Object[] getInvocationResults(Object ret,
Object[] parameters)
```

Merges out parameters and result of Workspace.invoke() into one array

**Parameters:**
- `ret` - Return value of Workspace.invoke()
- `parameters` - out parameters of the rpc call

**Returns:**
merged array

ggetHolderValue

```java
private Object getHolderValue(Object holder)
throws Exception
```

Gets the value of a holder object based on type introspection

**Parameters:**
- `holder` - Holder to get the value of

**Returns:**
the value of the holder

**Throws:**
*Exception*

---

Package org.dbe.toolkit.portal.interactionform

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfManager</td>
<td>130</td>
</tr>
</tbody>
</table>
class IfManager
extends Object

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>130</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfManager()</td>
<td>130</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getSM()</td>
<td>131</td>
</tr>
<tr>
<td>getSRI(String serventURL)</td>
<td>130</td>
</tr>
<tr>
<td>publish(String sm)</td>
<td>130</td>
</tr>
</tbody>
</table>

Field Detail

log

private static org.apache.log4j.Logger log

Constructor Detail

IfManager

public IfManager()

Method Detail

getSRI

public static org.dbe.kb.service.SRI getSRI(String serventURL)

Throws:
Exception

publish

public String publish(String sm)

throws Exception
Throws:
Exception

getSM

public String getSM()
Class SDLRetriever

logger

private static org.apache.log4j.Logger logger

helper

private org.sun.dbe.ClientHelper helper

Constructor Detail

SDLRetriever

public SDLRetriever(org.sun.dbe.ClientHelper helper)

Method Detail

getSDL

public String getSDL(String smid)
    throws org.dbe.servent.NoSuchServiceException, IOException

    Gets the SDL of the supplied SMID

    Parameters:
    
    smid - SMID to get the SDL of

    Returns:
    
    SDL as a string value

    Throws:
    
    org.dbe.servent.NoSuchServiceException
    IOException

Package org.dbe.toolkit.portal.qfsdt

Class Summary

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceSearcher</td>
<td>132</td>
</tr>
</tbody>
</table>

Class ServiceSearcher

org.dbe.toolkit.portal.qfsdt

java.lang.Object

Lorg.dbe.toolkit.portal.qfsdt.ServiceSearcher

public class ServiceSearcher

extends Object

Field Summary

<table>
<thead>
<tr>
<th>Field Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static static Hashtable modelElements</td>
<td>134</td>
</tr>
</tbody>
</table>
### Class ServiceSearcher

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>proxies</td>
<td>Hashtable</td>
<td>134</td>
</tr>
<tr>
<td>proxy</td>
<td>ProxyHandling</td>
<td>134</td>
</tr>
<tr>
<td>querySessionId</td>
<td>String</td>
<td>134</td>
</tr>
<tr>
<td>users</td>
<td>Hashtable</td>
<td>134</td>
</tr>
</tbody>
</table>

#### Constructor Summary

**ServiceSearcher ()**

#### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private void</td>
<td>addModelElement(Vector features)</td>
<td>136</td>
</tr>
<tr>
<td>void</td>
<td>closeSession(String sessionId)</td>
<td>136</td>
</tr>
<tr>
<td>Vector</td>
<td>convertRank(Vector results)</td>
<td>136</td>
</tr>
<tr>
<td>private org.dbe.kb.qi.adv.QueryExpr[]</td>
<td>createQueryExpr(Collection criteria)</td>
<td>137</td>
</tr>
<tr>
<td>org.dbe.kb.qi.adv.Template</td>
<td>createTemplate(String description, Collection templateElementPaths)</td>
<td>137</td>
</tr>
<tr>
<td>private Collection</td>
<td>getAsynchronousResults(String clientQuerySessionId)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getBMLModelFromId(String id)</td>
<td>137</td>
</tr>
<tr>
<td>private Vector</td>
<td>getChildren(AbstractNode node, Vector model)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getModelElement(String modelElementParentPath)</td>
<td>137</td>
</tr>
<tr>
<td>private Vector</td>
<td>getModelFromId(String modelId, String metamodel, String nodeId, boolean isSM)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getModelIds(String keywordsAndId)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getMoreResults(String clientQuerySessionId)</td>
<td>137</td>
</tr>
<tr>
<td>private String</td>
<td>getPath(AbstractNode node)</td>
<td>137</td>
</tr>
<tr>
<td>ProxyHandling</td>
<td>getProxy()</td>
<td>137</td>
</tr>
<tr>
<td>String</td>
<td>getServiceAvailabilityOnWeb(String id)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getServices(String keywordsAndId)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getSMInfo(String id)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>getSSLModelFromId(String id)</td>
<td>137</td>
</tr>
<tr>
<td>private Vector</td>
<td>getTokens(String keywords)</td>
<td>137</td>
</tr>
<tr>
<td>String</td>
<td>getURLToExecuteService(String id)</td>
<td>137</td>
</tr>
<tr>
<td>private void</td>
<td>manageSessions(String sessionId, boolean isNewQuery)</td>
<td>137</td>
</tr>
<tr>
<td>Vector</td>
<td>performAdvancedSearch(String arguments)</td>
<td>137</td>
</tr>
<tr>
<td>Object</td>
<td>performSearch(String description, Collection templateElementPaths, Collection criteria)</td>
<td>137</td>
</tr>
<tr>
<td>Object</td>
<td>performSearch(org.dbe.kb.qi.adv.Template template, Collection criteria)</td>
<td>137</td>
</tr>
<tr>
<td>private void</td>
<td>removeProxyEntry(String sessionId)</td>
<td>137</td>
</tr>
<tr>
<td>private Vector</td>
<td>reverseVector(Vector vector)</td>
<td>137</td>
</tr>
<tr>
<td>private Vector</td>
<td>setRoot(ContextRoot root, String modelId, String metamodel)</td>
<td>137</td>
</tr>
</tbody>
</table>
Field Detail

querySessionId
private String querySessionId

proxy
private ProxyHandling proxy

users
private static Hashtable users

proxies
private static Hashtable proxies

modelElements
private static Hashtable modelElements

Constructor Detail

ServiceSearcher
public ServiceSearcher()

Method Detail

performSearch
public Object performSearch(String description,
Collection templateElementPaths,
Collection criteria)

performSearch
public Object performSearch(org.dbe.kb.qi.adv.Template template,
Collection criteria)

createQueryExpr
private org.dbe.kb.qi.adv.QueryExpr[] createQueryExpr(Collection criteria)
Class ServiceSearcher

createTemplate

public org.dbe.kb.qi.adv.Template createTemplate(String description,
Collection templateElementPaths)

manageSessions

private void manageSessions(String sessionId,
boolean isNewQuery)

getServices

public Vector getServices(String keywordsAndId)

getModelIds

public Vector getModelIds(String keywordsAndId)

getSSLModelFromId

public Vector getSSLModelFromId(String id)

getBMLModelFromId

public Vector getBMLModelFromId(String id)

getSMInfo

public Vector getSMInfo(String id)

getAsynchronousResults

private Collection getAsynchronousResults(String clientQuerySessionId)

getMoreResults

public Vector getMoreResults(String clientQuerySessionId)

getModelFromId

private Vector getModelFromId(String modelId,
String metamodel,
String nodeId,
boolean isSM)
Class ServiceSearcher

setRoot

private Vector setRoot(ContextRoot root,
 String modelId,
 String metamodel)

getChildren

private Vector getChildren(AbstractNode node,
 Vector model)

addModelElement

private void addModelElement(Vector features)

getModelElement

public Vector getModelElement(String modelElementParentPath)

performAdvancedSearch

public Vector performAdvancedSearch(String arguments)

getURLToExecuteService

public String getURLToExecuteService(String id)

getTokens

private Vector getTokens(String keywords)

reverseVector

private Vector reverseVector(Vector vector)

getPath

private String getPath(AbstractNode node)

closeSession

public void closeSession(String sessionId)

removeProxyEntry

private void removeProxyEntry(String sessionId)
getProxy

public ProxyHandling getProxy() {
}

convertRank

public Vector convertRank(Vector results) {
}

getServiceAvailabilityOnWeb

public String getServiceAvailabilityOnWeb(String id) {
}

Package org.dbe.toolkit.portal.service

Interface Summary

<table>
<thead>
<tr>
<th>Interface Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBEPortal</td>
<td>137</td>
</tr>
</tbody>
</table>

Class Summary

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBEPortalAdapter</td>
<td>138</td>
</tr>
<tr>
<td>DBEPortalRegistrationClient</td>
<td>143</td>
</tr>
<tr>
<td>DBEPortalServiceConfig</td>
<td>148</td>
</tr>
<tr>
<td>SMIDGenerator</td>
<td>150</td>
</tr>
</tbody>
</table>

Interface DBEPortal

org.dbe.toolkit.portal.service

All Known Implementing Classes:

DBEPortalAdapter

public interface DBEPortal

Author:

andy-edmonds

Method Summary

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>void getUIURL(String smid, StringHolder url)</td>
<td>138</td>
</tr>
<tr>
<td>void getUrl(StringHolder url)</td>
<td>138</td>
</tr>
</tbody>
</table>
Method Detail

getURL

public void getURL(StringHolder url)
    throws RemoteException

Parameters:
    url - the return value - this is the URL where the portal is at

Throws:
    RemoteException
    java.rmi.RemoteException

getUIURL

public void getUIURL(String smid,
    StringHolder url)
    throws RemoteException

Parameters:
    smid - the SMID of the requested UI
    url - the return value - this is the URL where the UI is at

Throws:
    RemoteException
    java.rmi.RemoteException

Class DBEPortalAdapter

org.dbe.toolkit.portal.service

java.lang.Object
    org.dbe.toolkit.portal.service.DBEPortalAdapter

All Implemented Interfaces:
    org.dbe.servent.Adapter, DBEPortal

public class DBEPortalAdapter
extends Object
implements DBEPortal, org.dbe.servent.Adapter

Author:
    andy-edmonds

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private boolean configured</td>
<td>141</td>
<td></td>
</tr>
<tr>
<td>private context</td>
<td>141</td>
<td></td>
</tr>
<tr>
<td>private org.dbe.servent.ServiceContext context</td>
<td>141</td>
<td></td>
</tr>
<tr>
<td>private static String DEPLOY_ROOT</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>private static final String GET_SM</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>private static final String HTTP</td>
<td>140</td>
<td></td>
</tr>
</tbody>
</table>
### Class DBEPortalAdapter

<table>
<thead>
<tr>
<th>Private Static Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>String LOCALHOST</td>
</tr>
<tr>
<td>logger</td>
</tr>
<tr>
<td>String NIX_FILEPATH_SEP</td>
</tr>
<tr>
<td>String PORT_PREFIX</td>
</tr>
<tr>
<td>String PORTAL_CONTEXT</td>
</tr>
<tr>
<td>String PORTAL_PORT</td>
</tr>
<tr>
<td>String PORTAL_SM</td>
</tr>
<tr>
<td>boolean PROCESS_IF</td>
</tr>
<tr>
<td>String REL_UI_DEPLOY_DIR</td>
</tr>
<tr>
<td>String REL_UI_DROP_DIR</td>
</tr>
<tr>
<td>String UICACHE</td>
</tr>
</tbody>
</table>

#### Constructor Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBEPortalAdapter()</td>
<td>141</td>
</tr>
</tbody>
</table>

#### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>void destroy()</td>
<td>143</td>
</tr>
<tr>
<td>void getPort()</td>
<td>142</td>
</tr>
<tr>
<td>static org.apache.log4j.Logger getLogger()</td>
<td>143</td>
</tr>
<tr>
<td>void getUIURL(String smid, StringHolder url)</td>
<td>142</td>
</tr>
<tr>
<td>void getURL(StringHolder url)</td>
<td>141</td>
</tr>
<tr>
<td>void init(org.dbe.servent.ServiceContext context)</td>
<td>142</td>
</tr>
<tr>
<td>private void initialiseParameters()</td>
<td>142</td>
</tr>
<tr>
<td>private boolean isPortalPublished()</td>
<td>143</td>
</tr>
<tr>
<td>private void processIF(String smid)</td>
<td>143</td>
</tr>
<tr>
<td>private void setLoggingLevel()</td>
<td>142</td>
</tr>
</tbody>
</table>

#### Methods inherited from interface org.dbe.toolkit.portal.service.DBEPortal

- getUIURL
- getURL

#### Methods inherited from interface org.dbe.servent.Adapter

- destroy
- init
Field Detail

**NIX_FILEPATH_SEP**
private static final String NIX_FILEPATH_SEP

**LOCALHOST**
private static final String LOCALHOST

**PORT_PREFIX**
private static final String PORT_PREFIX

**HTTP**
private static final String HTTP

**EXTN**
private static final String EXTN

**GET_SM**
private static String GET_SM

**DEPLOY_ROOT**
private static String DEPLOY_ROOT

**UICACHE**
private static String UICACHE

**REL_UI_DEPLOY_DIR**
private static String REL_UI_DEPLOY_DIR

**REL_UI_DROP_DIR**
private static String REL_UI_DROP_DIR
Class DBEPortalAdapter

PORTAL_CONTEXT
private static String PORTAL_CONTEXT

PORTAL_SM
private static String PORTAL_SM

PROCESS_IF
private static boolean PROCESS_IF

custom
private org.dbe.servent.ServiceContext context

configured
private boolean configured

PORTAL_PORT
private static String PORTAL_PORT

logger
private static org.apache.log4j.Logger logger

Constructor Detail

DBEPortalAdapter
public DBEPortalAdapter()

Method Detail

getURL
public void getURL(StringHolder url)
throws RemoteException

Gets the URL where the portal runs at
Specified by:
getURL in interface DBPortal
Parameters:
url - the return value - the portal's URL
Class DBEPortalAdapter

Throws:
   RemoteException

getPort

private String getPort()

Gets the port on which the portal runs on
Returns:
   the port number

getUIURL

public void getUIURL(String smid,
    StringHolder url)
    throws RemoteException

Zips up the requested UI (corresponding to the SMID) and returns back the URL of where it is available
Specified by:
   getUIURL in interface DBEPortal
Parameters:
   smid - the SMID of the service which implements the requested UI
   url - a return value - the URL of the zip file containing the UI
Throws:
   RemoteException

init

public void init(org.dbe.servent.ServiceContext context)

Initialises the PortalService
Specified by:
   init in interface org.dbe.servent.Adapter
Parameters:
   context - the service context containing runtime information of this service

setLoggingLevel

private void setLoggingLevel()

Sets the appropriate logging level from the portal service configuration file TODO replace this with a configuration from log4j.properties file

initialiseParameters

private void initialiseParameters()
Class DBEPortalAdapter

Sets the parameters associated with this service. Called by the init method of the service

isPortalPublished

private boolean isPortalPublished() throws IOException

Publishes the service to the SR only if it has never been Called only once when the service runs for the first time

Throws:
	IOException

destroy

public void destroy()

Called when the service is shutdown

Specified by:
	destroy in interface org.dbe.servent.Adapter

processIF

private void processIF(String smid)

Parameters:
	smid - the Service manifest ID

Author:
	mbordin

getLogger

public static org.apache.log4j.Logger getLogger()
### Constructor Summary

- **DBEPortalRegistrationClient**(org.dbe.servent.ServiceContext svcCtx)

### Method Summary

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.w3c.dom.Document createXMLDoc(String xmlFile)</td>
<td>Reads in a file containing XML and creates a DOM of it's content.</td>
<td>147</td>
</tr>
<tr>
<td>private void doSMEDataEntry(String serviceManisfestPath)</td>
<td>Enters the BML Data into the PortalService's SM</td>
<td>147</td>
</tr>
<tr>
<td>private void doSMEPortalPublish(String SMID, String serviceManifestPath)</td>
<td>Publishes the SM to the SR with the supplied SMID</td>
<td>148</td>
</tr>
<tr>
<td>static void main(String[] args)</td>
<td>Publishes the PortalService to the SR along with the BML Data</td>
<td>147</td>
</tr>
<tr>
<td>private org.w3c.dom.Document replaceAllXMLAttributeValue(String xmlFile, String elementName, HashMap replacementValues)</td>
<td>Replaces the value of the specified element in an XML file</td>
<td>147</td>
</tr>
<tr>
<td>private void setIsPublished(String serviceManifestPath)</td>
<td></td>
<td>148</td>
</tr>
<tr>
<td>void setProps(Properties props1)</td>
<td></td>
<td>148</td>
</tr>
<tr>
<td>void setSM_FILE(String sm_file)</td>
<td></td>
<td>148</td>
</tr>
<tr>
<td>private String smToString(String serviceManifestPath)</td>
<td>Converts the supplied file to a string</td>
<td>148</td>
</tr>
<tr>
<td>void writeXmlFile(org.w3c.dom.Document doc, String filename)</td>
<td>Writes a XML DOM to file.</td>
<td>147</td>
</tr>
</tbody>
</table>
### Field Detail

**SERVENT_HOME**

private static String SERVENT_HOME

---

**SM_FILE**

private static String SM_FILE

---

**SR**

private static final String SR

---

**IMM_DATAVALUE**

private static final String IMM_DATAVALUE

---

**XMI_ID**

private static final String XMI_ID

---

**VALUE**

private static final String VALUE

---

**DESCRIPTION_ATTR**

private static final String DESCRIPTION_ATTR

---

**BIZ_DOM_ATTR**

private static final String BIZ_DOM_ATTR

---

**COUNTRY_ATTR**

private static final String COUNTRY_ATTR

---

**REGION_ATTR**

private static final String REGION_ATTR
class DBEPortalRegistrationClient

LOCALITY_ATTR
private static final String LOCALITY_ATTR

PORTAL_DESCR
private static final String PORTAL_DESCR

PORTAL_BIZDOM
private static final String PORTAL_BIZDOM

PORTAL_PAYS
private static final String PORTAL_PAYS

PORTAL_REGION
private static final String PORTAL_REGION

PORTAL_LOC
private static final String PORTAL_LOC

svcCtx
private org.dbe.servent.ServiceContext svcCtx

props
private static Properties props

logger
private org.apache.log4j.Logger logger

Constructor Detail

DBEPortalRegistrationClient

public DBEPortalRegistrationClient (org.dbe.servent.ServiceContext svcCtx)
    throws Exception

Constructor
Class DBEPortalRegistrationClient

Throws:
   Exception

Method Detail

publish

public void publish(String serviceManifestPath)
   throws Exception

   Publishes the PortalService to the SR along with the BML Data
   Throws:
      Exception

doSMEDataEntry

private void doSMEDataEntry(String serviceManifestPath)

   Enters the BML Data into the PortalService's SM

replaceAllXMLAttributeValue

private org.w3c.dom.Document replaceAllXMLAttributeValue(String xmlFile,
   String elementName,
   HashMap replacementValues)

   Replaces the value of the specified element in an XML file
   Returns:
      the modified xml document

createXMLDoc

public org.w3c.dom.Document createXMLDoc(String xmlFile)
   throws FactoryConfigurationException

   Reads in a file containing XML and creates a DOM of it's content.
   Returns:
      the xml DOM
   Throws:
      FactoryConfigurationException

writeXmlFile

public void writeXmlFile(org.w3c.dom.Document doc,
   String filename)

   Writes a XML DOM to file.
   Parameters:
      filename - the location where the file should be written to
Class DBEPortalRegistrationClient

**doSMEPortalPublish**

```java
private void doSMEPortalPublish(String SMID,
    String serviceManifestPath)
```

Publishes the SM to the SR with the supplied SMID

**Parameters:**

- `SMID` - the SMID that is to be published with the SM

**smToString**

```java
private String smToString(String serviceManifestPath)
```

Converts the supplied file to a string

**setIsPublished**

```java
private void setIsPublished(String serviceManifestPath)
```

**setProps**

```java
public void setProps(Properties props1)
```

**setSM_FILE**

```java
public void setSM_FILE(String sm_file)
```

**main**

```java
public static void main(String[] args)
```

Class DBEPortalServiceConfig

```java
public class DBEPortalServiceConfig
```

**Field Summary**

<table>
<thead>
<tr>
<th>Field</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static String CONFIG_FILE</td>
<td>149</td>
</tr>
<tr>
<td>private static Properties portalServiceConfig</td>
<td>149</td>
</tr>
</tbody>
</table>
Class DBEPortalServiceConfig

Constructor Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>DBEPortalServiceConfig()</td>
<td>Constructor</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static Properties</td>
<td>getPortalServiceConfig()</td>
<td>Gets portal service configuration</td>
</tr>
<tr>
<td>static String</td>
<td>getString(String key)</td>
<td>Gets a string from configuration</td>
</tr>
<tr>
<td>static void</td>
<td>setCONFIG_FILE(String config_file)</td>
<td>Sets the configuration file</td>
</tr>
<tr>
<td>static void</td>
<td>setPortalServiceConfig(Properties portalServiceConfig)</td>
<td>Sets the portal service configuration</td>
</tr>
<tr>
<td>static void</td>
<td>store(String path)</td>
<td>Stores configuration to file</td>
</tr>
</tbody>
</table>

Field Detail

portalServiceConfig

private static Properties portalServiceConfig

CONFIG_FILE

private static String CONFIG_FILE

Constructor Detail

DBEPortalServiceConfig

private DBEPortalServiceConfig()

Method Detail

getString

public static String getString(String key) 

Throws: 

IOException

getPortalServiceConfig

public static Properties getPortalServiceConfig()

store

public static void store(String path)
Class DBEPortalServiceConfig

setCONFIG_FILE

public static void setCONFIG_FILE(String config_file)

setPortalServiceConfig

public static void setPortalServiceConfig(Properties portalServiceConfig)

Class SMIDGenerator

org.dbe.toolkit.portal.service

java.lang.Object
  org.dbe.toolkit.portal.service.SMIDGenerator

public class SMIDGenerator
  extends Object

Field Summary

private static org.apache.log4j.Logger logger

Constructor Summary

SMIDGenerator()
  Default constructor

Method Summary

private org.w3c.dom.Document createXMLDoc(String xmlFile)

String getUID()
  Generates a unique ID

static void main(String[] args)

private void printUsage()
  Prints the command line usage

private org.w3c.dom.Document replaceXMLElementValue(String xmlFile, String elementName, String newElementValue)

void writeSMIDtoFile(String smid, String serviceDeployConf)
  Writes a SMID to the services deployment descriptor

void writeXmlFile(org.w3c.dom.Document doc, String filename)

Field Detail

logger

private static org.apache.log4j.Logger logger
Class SMIDGenerator

Constructor Detail

SMIDGenerator

public SMIDGenerator()

Default constructor

Method Detail

printUsage

private void printUsage()

Prints the command line usage

getUID

public String getUID()

Generates a unique ID

Returns:
the unique ID as a string

writeSMIDToFile

public void writeSMIDToFile(String smid,
String serviceDeployConf)

Writes a SMID to the services deployment descriptor

Parameters:
 serviceDeployConf - the location of the service's deployment descriptor

replaceXMLElementValue

private org.w3c.dom.Document replaceXMLElementValue(String xmlFile,
String elementName,
String newElementValue)

throws FactoryConfigurationError

Throws:
 FactoryConfigurationError

createXMLDoc

private org.w3c.dom.Document createXMLDoc(String xmlFile)

throws FactoryConfigurationError

Throws:
 FactoryConfigurationError
Class SMIDGenerator

writeXmlFile

public void writeXmlFile(org.w3c.dom.Document doc, String filename)

main

public static void main(String[] args)

Package org.dbe.toolkit.portal.ui

Class Summary

<table>
<thead>
<tr>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>UICache</td>
<td>152</td>
</tr>
<tr>
<td>UIRetrievalDelegate</td>
<td>154</td>
</tr>
</tbody>
</table>

Class UICache

org.dbe.toolkit.portal.ui

java.lang.Object

org.dbe.toolkit.portal.ui.UICache

public class UICache
extends Object

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean cacheEnabled</td>
<td>153</td>
</tr>
<tr>
<td>HashMap uiCache</td>
<td>153</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>UICache ()</td>
<td>153</td>
</tr>
<tr>
<td>UICache(String cacheDirectory)</td>
<td>153</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>String get(String smid)</td>
<td>153</td>
</tr>
<tr>
<td>private HashMap getUiCache()</td>
<td>153</td>
</tr>
<tr>
<td>boolean isCached(String smid)</td>
<td>153</td>
</tr>
<tr>
<td>boolean isCacheEnabled()</td>
<td>153</td>
</tr>
<tr>
<td>void setCacheEnabled(boolean cacheEnabled)</td>
<td>153</td>
</tr>
<tr>
<td>void store(String smid, String ui)</td>
<td>153</td>
</tr>
</tbody>
</table>
Class UICache

Field Detail

uiCache
HashMap uiCache

cacheEnabled
boolean cacheEnabled

Constructor Detail

UICache
public UICache()

UICache
public UICache(String cacheDirectory)

Method Detail

isCached
public boolean isCached(String smid)

store
public void store(String smid,
                   String ui)

getUiCache
private HashMap getUiCache()

get
public String get(String smid)

isCacheEnabled
public boolean isCacheEnabled()

setCacheEnabled
public void setCacheEnabled(boolean cacheEnabled)
Class UIRetrievalDelegate

public class UIRetrievalDelegate extends Object

Author: andy-edmonds

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static String EXTN</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private org.sun.dbe.ClientHelper helper</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String HTTP</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String HTTP_LOCALHOST_80_PORTAL_UICACHE</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private boolean initialized</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static org.apache.log4j.Logger logger</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String NIX_FILEPATH_SEP</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String PORT_PREFIX</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String PORTAL_DEFAULT_PORTAL_ID</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>private static String PORTAL_DEFAULT_UI_ENTRY_POINT</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>private static String PORTAL_GET_UIURL</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>private static String PORTAL_HOST</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>private static String PORTAL_PORT</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String PORTAL_UI_CACHE_CONTEXT</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String PORTAL_UI_CACHE_DIR</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>private static String SERVENT_DEPLOY_DIR</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>private org.dbe.studio.tools.smcreator.core.eb.SM sm</td>
<td></td>
<td>157</td>
</tr>
<tr>
<td>private org.dbe.studio.tools.smcreator.core.SMCreator smc</td>
<td></td>
<td>157</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>UIRetrievalDelegate(org.sun.dbe.ClientHelper helper)</td>
<td>157</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getCacheLocalUI(String smid)</td>
<td>Downloads the remote UI as a zip file, extracts it and generates a URL for it</td>
<td>158</td>
</tr>
<tr>
<td>getLocalUIURL(File outputFile, String smid)</td>
<td>Unzips the received remote UI zip, extracts it to a web context and returns back the URL of that locally cached UI</td>
<td>159</td>
</tr>
<tr>
<td>getPortalEndpoint(String smid)</td>
<td></td>
<td>157</td>
</tr>
</tbody>
</table>
private String getPortalID(String smid)
Gets the SMID of the portal where the requested UI is located 159

private String getServiceUI(String smid)
This gets the URL where the service UI is located 160

String getUI(String smid)
Gets the URL of a locally cached copy of the requested UI 157

private File getUIArchive(String smid, String uiZipURL)
Downloads the zipped UI from the remote servant 159

private String getUIArchiveURL(String smid)
Gets the URL of the remote UI zip file 158

private String getUIEntryPoint(String url)
This gets the main entry point for an openlaszlo UI 160

private String getUIIF(String smid) 158

private orgdbe.toolkit.proxyframework.Workspace getWorkspace(String smid)
Gets a Workspace object 160

private boolean isIF(String smid) 158

private void initParams()
Initialises parameters from the portalClientConfig.properties file 157

private boolean isPortal(String smid) 158

Field Detail

SERVENT_DEPLOY_DIR
private static String SERVENT_DEPLOY_DIR

PORTAL_HOST
private static String PORTAL_HOST

PORTAL_DEFAULT_UI_ENTRY_POINT
private static String PORTAL_DEFAULT_UI_ENTRY_POINT

PORTAL_DEFAULT_PORTAL_ID
private static String PORTAL_DEFAULT_PORTAL_ID

PORTAL_GET_UIURL
private static String PORTAL_GET_UIURL
Class UIRetrievalDelegate

PORTAL_UI_CACHE_DIR
private static String PORTAL_UI_CACHE_DIR

PORTAL_UI_CACHE_CONTEXT
private static String PORTAL_UI_CACHE_CONTEXT

PORTAL_PORT
private static String PORTAL_PORT

HTTP_LOCALHOST_80_PORTAL_UICACHE
private static String HTTP_LOCALHOST_80_PORTAL_UICACHE

HTTP
private static String HTTP

EXTN
private static String EXTN

PORT_PREFIX
private static String PORT_PREFIX

NIX_FILEPATH_SEP
private static String NIX_FILEPATH_SEP

helper
private org.sun.dbe.ClientHelper helper

initialised
private boolean initialised

logger
private static org.apache.log4j.Logger logger
Class UIRetrievaleDelegate

sm

private org.dbe.studio.tools.smcreator.core.eb.SM sm

Author: mbordin This variable it is needed to IF serviceManifest

smc

private org.dbe.studio.tools.smcreator.core.SMCreator smc

Constructor Detail

UIRetrievaleDelegate

public UIRetrievaleDelegate(org.sun.dbe.ClientHelper helper)

Method Detail

initParams

private void initParams()

Initialises parameters from the portalClientConfig.properties file

getUI

public String getUI(String smid)
   throws DBEPortalException

   Gets the URL of a locally cached copy of the requested UI
   Parameters:
      smid - the SMID of the service which has the requested UI
   Returns:
      the URL of the UI
   Throws:
      DBEPortalException

getPortalEndpoint

private String getPortalEndpoint(String smid)
   throws DBEPortalException

   Throws:
      DBEPortalException
Class UIRetrievalDelegate

isPortal

private boolean isPortal(String smid)

getUIIF

private String getUIIF(String smid)
   throws Exception

   Parameters:
   smid - it is the ServiceManifestID

   Returns:
   String It return the url to lunch

   Throws:
   Exception - If something it is wrong it throw an exception the method create the correct UI
   for an IF serviceManifest

Author:
mbordin

ifisIF

private boolean ifisIF(String smid)

   Parameters:
   smid - It is the service manifest ID

   Returns:
   boolean It return TRUE if a serviceManifest it is an iteration form.

Author:
mbordin

getCacheLocalUI

private String getCacheLocalUI(String smid)
   throws DBEPortalException

   Downloads the remote UI as a zip file, extracts it and generates a URL for it

   Parameters:
   smid - SMID of the service implementing the requested UI

   Returns:
   URL of the locally cached UI

   Throws:
   DBEPortalException

getUIArchiveURL

private String getUIArchiveURL(String smid)
   throws DBEPortalException

   Gets the URL of the remote UI zip file
Class UIRetrievalDelegate

getPortalID

private String getPortalID(String smid)
throws DBEPortalException

gets the SMID of the portal where the requested UI is located

Parameters:

  smid - SMID of the service implementing the UI requested

Returns:

  the ID of the hosting portal

Throws:

  DBEPortalException

getUIArchive

private File getUIArchive(String smid, String uiZipURL)
throws DBEPortalException

Downloads the zipped UI from the remote server

Parameters:

  smid - SMID of the service that implements the UI
  uiZipURL - the URL of the zipped UI

Returns:

  a

Throws:

  DBEPortalException

See Also:

  object that corresponds to the zip file containing the UI

getLocalUIURL

private String getLocalUIURL(File outputFile, String smid)
throws DBEPortalException

Unzips the received remote UI zip, extracts it to a web context and returns back the URL of that locally cached UI

Parameters:

  outputFile - the file to unzip
  smid - the SMID of the service implementing the UI

Returns:

  the local URL of the UI

Throws:

  DBEPortalException
getServiceUI

private String getServiceUI(String smid)

This gets the URL where the service UI is located

Parameters:
  smid - SMID of the service implementing the UI

Returns:
  The URL of the UI

getUIEntryPoint

private String getUIEntryPoint(String url)

This gets the main entry point for an openlaszlo UI

Parameters:
  url - The URL to extract the main entry point from

Returns:
  The main entry point

getWorkspace

private org.dbe.toolkit.proxyframework.Workspace getWorkspace(String smid)
  throws DBEPortalException

Gets a Workspace object

Returns:
  The workspace object

Throws:
  DBEPortalException

Package org.dbe.toolkit.portal.ui.tools
Class UIResourcesUnzipper

Author:
  andy-edmonds

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static org.apache.log4j.Logger logger</td>
<td>161</td>
</tr>
<tr>
<td>private static final String NIX_FILEPATH_SEP</td>
<td>161</td>
</tr>
</tbody>
</table>

Constructor Summary

UIResourcesUnzipper()

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>private static org.apache.log4j.Logger getLogger()</td>
<td>162</td>
</tr>
<tr>
<td>static void unzip(String zipName, String extractionPath) Extracts a zip file</td>
<td>161</td>
</tr>
</tbody>
</table>

Field Detail

NIX_FILEPATH_SEP

private static final String NIX_FILEPATH_SEP

logger

private static org.apache.log4j.Logger logger

Constructor Detail

UIResourcesUnzipper

public UIResourcesUnzipper()

Method Detail

unzip

public static void unzip(String zipName, String extractionPath) throws IOException

Extracts a zip file

Parameters:
  zipName - the absolute name of the zip file
  extractionPath - the absolute path to where the zip file is to be extracted to

Throws:
  IOException
getLogger

private static org.apache.log4j.Logger getLogger()
Class UIResourcesZipper

org.dbe.toolkit.portal.ui.tools

java.lang.Object

    org.dbe.toolkit.portal.ui.tools.UIResourcesZipper

public class UIResourcesZipper
    extends Object

Author:
    andy-edmonds

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIX_FILEPATH_SEP</td>
<td>private static final String</td>
<td>163</td>
</tr>
<tr>
<td>relative</td>
<td>private static boolean</td>
<td>163</td>
</tr>
<tr>
<td>relativeDir</td>
<td>private static String</td>
<td>163</td>
</tr>
</tbody>
</table>

Constructor Summary

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>UIResourcesZipper()</td>
<td>164</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>setRelative(boolean relative)</td>
<td>static void</td>
<td>164</td>
</tr>
<tr>
<td>zip(org.apache.log4j.Logger logger, File[] files, ZipOutputStream zos)</td>
<td>static void</td>
<td>164</td>
</tr>
<tr>
<td>zip(org.apache.log4j.Logger logger, String file, String saveTo)</td>
<td>static void</td>
<td>164</td>
</tr>
</tbody>
</table>

Field Detail

NIX_FILEPATH_SEP
private static final String NIX_FILEPATH_SEP

relative
private static boolean relative

relativeDir
private static String relativeDir
Class UIResourcesZipper

Constructor Detail

UIResourcesZipper

public UIResourcesZipper()

Method Detail

zip

public static void zip(org.apache.log4j.Logger logger,
                    String file,
                    String saveTo)
               throws IOException

create a zip file from a list of files it'll recursively zip if a file's a directory

Parameters:
   file - Name of the zip file to create
   saveTo - location to save the created zip file to

Throws:
   IOException
   Exception

zip

private static void zip(org.apache.log4j.Logger logger,
                   File[] files,
                   ZipOutputStream zos)
              throws IOException

   Throws:
   IOException

setRelative

public static void setRelative(boolean relative)

Sets archive creation to 'relative'

Parameters:
   relative - a boolean to set the flag