OKI Developers Meeting

2/14/2002 — 2/19/2002

Key West, Florida
OKI Developers,

Thank you for attending this meeting. We are looking forward to seeing you in Key West. This meeting is the last to focus on the Common Services layer and the first to focus on the OKI functional layer.

We expect to spend roughly the first half of the meeting on the Common Services layer and the second half on the OKI functional layer.

Some of the purposes of this meeting are to:
- Assess the current state of the OKI architecture
- Evaluate the OKI Common Service APIs
- Understand example implementations
- Decide various naming conventions
- Plan next steps for OKI Common Service implementations
- Develop the OKI functional layer concepts

Attached is a current snapshot of the OKI code. We will have hard copy for the meeting. The OKI Common Service APIs are going to be reviewed in detail so that we finalize them. The implementation code is there for reference and as examples. The rest including the okifuncapi are experimental.

Here are some possible topics for the meeting. Please think about other topics you would like to discuss.
- Various approaches to implementing Factories
- How OKI applications use the APIs
- Examination of the authen api
- Location of reference implementations
- API naming alternatives — for example:
  - Should it be authen or authN or ?
  - Should it be roles or authZ or ?
  - Should it be rmi or ormi or ?
  - Should it be sql or osql or ?
- OKI Assets, Tools and Associations
- CVS structure and access
- API review, release, and versioning procedures
- OKI License and Copyright
- Demos of other efforts
- Development and platform discussion
- Documentation, educational materials for OKI
- Communication, coordination web site

See you in Key West,

MIT OKI developers
package org.okiproject.serviceapi.authen;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public class AuthenException extends java.lang.Exception {
    public static final String UNIMPLEMENTED = "Unimplemented Authen method ";
    public static final String DEFAULTAUTHENFACTORYDIDNOTLOAD = "Default Authen factory did not load ";
    public static final String AUTHENFACTORYPROPERTIESDIDNOTLOAD = "Authen factory properties did not load ";

    /**
     * Constructor AuthenException
     *
     *
     * @param message
     *
     */
    public AuthenException(String message) {
        super(message);
    }
}
package org.okiproject.serviceapi.authen;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

   The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

**/

The Authen API does not contain information that is expected to be part of the implementation. There may be additional implementation information that will need to be exposed in this API.

*/

public class AuthenFactory

implements java.io.Serializable

{  
    private static final org.okiproject.serviceapi.authen.AuthenException UNIMPLEMENTED =
        new org.okiproject.serviceapi.authen.AuthenException(
            org.okiproject.serviceapi.authen.AuthenException.UNIMPLEMENTED);

    /**
     * Method getDefaultAuthenFactory
     *
     * @param agent
     * @return org.okiproject.serviceapi.authen.AuthenFactory
     * @throws Exception
     */

    public static org.okiproject.serviceapi.authen.AuthenFactory getDefaultAuthenFactory(
        org.okiproject.serviceapi.common.Agent agent, java.util.Map props)

        throws Exception
    {
        java.util.Map properties = props;
        if (null == properties) {
            properties = getPropertiesFromPRB();
        } else if (null != properties) {
            String factoryClassName = (String) properties.get("AuthenFactory");
            if (null != factoryClassName) {
                Class                                          factoryClass =          Class.forName(factoryClassName);        factory      =          (org.okiproject.serviceapi.authen.AuthenFactory) factoryClass.newInstance();        factory.setAgent(agent);

        factory.setProperties(properties);
        return factory;
    }

    throw new org.okiproject.serviceapi.authen.AuthenException(
            org.okiproject.serviceapi.authen.AuthenException.DEFAULTAUTHENFACTORYDIDNOTLOAD);

    } //

private org.okiproject.serviceapi.common.Agent agent = null;

protected void setAgent(org.okiproject.serviceapi.common.Agent agent) {
    this.agent = agent;
}

private java.util.Map properties = null;

protected java.util.Map getProperties() {
    return this.properties;
}

protected void setProperties(java.util.Map properties) {
    this.properties = properties;
}

protected static java.util.Properties getPropertiesFromPRB()

    throws Exception
    {
        java.util.PropertyResourceBundle prb =
            {java.util.PropertyResourceBundle} java.util.ResourceBundle.getBundle("AuthenFactory");

    }

} //
if (null != prb) {
  java.util.Properties properties = new java.util.Properties();
  java.util.Enumeration keys = prb.getKeys();
  while (keys.hasMoreElements()) {
    String key = (String) keys.nextElement();
    properties.setProperty(key, (String) prb.handleGetObject(key));
  }
  return properties;
}
throw new org.okiproject.serviceapi.authen.AuthenException{
  org.okiproject.serviceapi.authen.AuthenException.AUTHENFACTORYPROPERTIESDIDNOTLOAD
} ;

public org.okiproject.serviceapi.common.Agent getAgent()
throws Throwable {
  throw UNIMPLEMENTED;
}

public String[] getAuthenTypes()
throws Exception {
  throw UNIMPLEMENTED;
}

public String[] getAuthenTypes(org.okiproject.serviceapi.common.Agent agent)
throws Exception {
  throw UNIMPLEMENTED;
}

/**
 * @return org.okiproject.serviceapi.common.Agent
 * @throws Throwable
 */
public org.okiproject.serviceapi.common.Agent getAgent(
  org.okiproject.serviceapi.common.Agent agent, String type)
throws Throwable {
  throw UNIMPLEMENTED;
}

/**
 * @Method destroyAuthen
 * @param agent
 * @param type
 * @throws Exception
 */
public void destroyAuthen(org.okiproject.serviceapi.common.Agent agent)
throws Exception {
  throw UNIMPLEMENTED;
}

/**
 * @Method destroyAuthen
 * @param agent
 * @param type
 * @throws Exception
 */
public void destroyAuthen(org.okiproject.serviceapi.common.Agent agent,
  String[] types)
throws Exception {
  throw UNIMPLEMENTED;
}
package org.okiproject.serviceapi.common;

/**
Copyright (c) 2001 Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

/**
* Purpose: Key Abstraction representing an entity interacting with the service
* APIs.
* $Source: /cvs/oki/org/okiproject/serviceapi/common/Agent.java,v $
* $Revision: 1.3 $
* $Date: 2002/02/08 14:25:42 $
*******************************************************************************/

public interface Agent
extends java.io.Serializable {

/**
 * Method getID
 */
Object getID();

/**
 * Method getName
 */
String getName();
}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public interface Agents extends java.io.Serializable {

/**
   * Method hasMoreAgents
   * @throws Exception
   */
   boolean hasMoreAgents() throws Exception;

/**
   * Method nextAgent
   * @throws Exception
   */
   org.okiproject.serviceapi.common.Agent nextAgent() throws Exception;
}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software
 * and associated documentation files (the "Software"), to deal in the Software without restriction,
 * including without limitation the rights to use, copy, modify, merge, publish, distribute,
 * sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or
 * substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
 * INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE
 * FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR
 * OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
 * DEALINGS IN THE SOFTWARE.
 ***/

public interface AuthorizableEntity extends Agent {

/**
 * Method getAuthenTypes
 *
 *
 */

String[] getAuthenTypes();}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 *
 * ************************************************************************** *
 * Purpose: Key Abstraction representing an Authorizable entity
 * interacting with the service APIs.
 * $Source: /cvs/oki/org/okiproject/serviceapi/common/Group.java,v $ 
 * $Revision: 1.3 $ 
 * $Date: 2002/02/08 14:25:42 $ 
 * ************************************************************************** *
 * /
 * public interface Group
 * extends AuthorizableEntity {}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

================================================================================

* Purpose: Key Abstraction representing an Authorizable entity
* interacting with the service APIs.
* $Source: /cvs/oki/org/okiproject/serviceapi/common/Machine.java,v $
* $Revision: 1.3 $
* $Date: 2002/02/08 14:25:42 $
================================================================================

public interface Machine
extends AuthorizableEntity {}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software,
 * and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions
 * of the Software.
 */

public interface NonAuthorizableEntity extends Agent {}
org/okiproject/serviceapi/common/Organization.java

package org.okiproject.serviceapi.common;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology

   Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

   The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

*/

public interface Organization extends Agent {}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Person extends AuthorizableEntity {

/** Method getID */
Object getID();

/** Method setID */
void setID(Object id);

/** Method getName */
String getName();

/** Method getProperties */
java.util.Properties getProperties();
}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

import org.okiproject.serviceapi.common.AuthorizableEntity;

public interface Service extends AuthorizableEntity {}
package org.okiproject.serviceapi.common;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Tool
extends AuthorizableEntity {}
/**
   * Copyright (c) 2001 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any
   * person obtaining a copy of this software and associated
   * documentation files (the "Software"), to deal in the
   * Software without restriction, including without limitation
   * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
   * Software, and to permit persons to whom the Software is
   * furnished to do so, subject to the following conditions:
   * The above copyright notice and this permission notice
   * shall be included in all copies or substantial portions of
   * the Software.
   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
   * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
   * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
   * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
   * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
   * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
   * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
   * IN THE SOFTWARE.
   */

public interface User
extends Person {}
package org.okiproject.serviceapi.file;

/*

Copyright (c) 2001 Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

/**
The ByteStore is the key interface. It is a set of bytes with associated
physical metadata.
If the Cabinet implementation supports it, each ByteStore may be
stored redundantly on multiple servers for protection against data
loss.
If the Cabinet implementation supports versioning, it may have several
versions of arbitrary depth.
For ease of use, the ByteStore interface parallels when appropriate the
java.io.File class.
Versioning may be supported in the Cabinet. The default is to return
the latest version of a ByteStore. Other versions may be accessed by a key
string. Each version is itself a ByteStore, and may have versions of its own, permitting branching. The underlying principle is that
all versions should serve the same basic purpose, have the same
mime-type, be functionally equivalent. If idiosyncrasies of a
particular version are significant, that version key may be
specified. When a version key is specified, the base version rather
than any latest sub-version is returned. Sub versions may then be
obtained if desired. Different versions will only differ in bytes of
content and therefore length, lastModified time, and digest/checksum.
A particular version may be given a GUID only by creating a new
ByteStore in the Cabinet.
*/

public interface ByteStore
extends java.io.Serializable {
  // from java.io.File
  /**
   * Method length
   *
   * @return long
   */
  public long length();

  /**
   * Method lastModified
   *
   * @return long
   */
  public long lastModified();

  /**
   * Method setLastModified
   *
   * @param time
   * @return boolean
   */
  public boolean setLastModified(long time);

  /**
   * Method deleteOnExit
   *
   */
  public void deleteOnExit();

  /**
   * Method toString
   *
   * @return String
   */
  public String toString();

  /**
   * Method canRead
   *
   * @return boolean
   */
  public boolean canRead();

  /**
   * Method canWrite
   *
   * @return boolean
   */
  public boolean canWrite();
/**
 * Method setReadOnly
 * @return boolean
 */
public boolean setReadOnly();

/**
 * Method setWritable
 * @return boolean
 */
public boolean setWritable(); // not in File, but seems needed?

/**
 * Method compareTo
 * @param o
 * @return int
 */
public int compareTo(Object o);

/**
 * Method equals
 * @param obj
 * @return boolean
 */
public boolean equals(Object obj);

/**
 * Method hashCode
 * @return int
 */
public int hashCode();

// added for OKI
/**
 * Method getGUID
 * @return org.okiproject.serviceapi.guid.GUID
 */
public org.okiproject.serviceapi.guid.GUID getGUID();

/**
 * Method getByteStoreName
 * @return String
 */
public String getByteStoreName();

/**
 * Method renameTo
 * @param newName
 * @return boolean
 */
public boolean renameTo(String newName);

/**
 * Method getMimeType
 * @return java.lang.String
 */
public java.lang.String getMimeType(); // Content-Type in jar file manifest

/**
 * Method setMimeType
 * @param mimeType
 */
public void setMimeType(java.lang.String mimeType);

// e.g. md5 or crc
/**
 * Method getDigest
 * @param digestAlgorithm
 * @return java.lang.String
 */
public java.lang.String getDigest(String digestAlgorithm);

// primarily for quota
/**
 * Method getAttributes
 * @return java.util.jar.Attributes
 */
public java.util.jar.Attributes getAttributes();

/**
 * Method getOwner
 * @return org.okiproject.serviceapi.common.Agent
 */
import org.okiproject.serviceapi.common.Agent;  

public org.okiproject.serviceapi.common.Agent getOwner();  
/**  
 * Method setOwner  
 *  
 * @param owner  
 *  
 */  
public void setOwner(org.okiproject.serviceapi.common.Agent owner);  
// for ease of use for i/o  
/**  
 * Method getInputStream  
 *  
 * @return java.io.InputStream  
 *  
 */  
public java.io.InputStream getInputStream();  
/**  
 * Method getOutputStream  
 *  
 * @return java.io.OutputStream  
 *  
 */  
public java.io.OutputStream getOutputStream();  
/**  
 * Method getReader  
 *  
 * @return java.io.Reader  
 *  
 */  
public java.io.Reader getReader();  
/**  
 * Method getWriter  
 *  
 * @return java.io.Writer  
 *  
 */  
public java.io.Writer getWriter();  
// built on basic read() and write() methods  
// versioning:  
/**  
 * Method hasVersions  
 *  
 * @return boolean  
 *  
 */  
public boolean hasVersions();  
/**  
 * Method getLatestVersion  
 *  
 * @return ByteStore  
 *  
 */  
public ByteStore getLatestVersion();  
/**  
 * Method getBaseVersion  
 *  
 * @return ByteStore  
 *  
 */  
public ByteStore getBaseVersion();  
/**  
 * Method getVersion  
 *  
 * @return ByteStore  
 *  
 */  
public ByteStore getLatestVersion(java.lang.String versionKey);  
/**  
 * Method getVersions  
 *  
 * @return ByteStore[]  
 *  
 */  
public ByteStore[] getVersions();  
/**  
 * Method newVersion  
 *  
 * @param agent  
 *  
 */  
public ByteStore newVersion(org.okiproject.serviceapi.common.Agent agent,  
String versionKey);  
}
package org.okiproject.serviceapi.file;

/*
   Copyright (c) 2001 Massachusetts Institute of Technology

   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

/**
Cabinets have properties, and contain other Cabinets and ByteStores.
They may manage quotas, that is, if the implementation supports
quotas, each Agent may be assigned a quota of space used in the
Cabinet.

Cabinets contain CabinetEntries, each of which may be a ByteStore or a
Cabinet. They are known by their GUIDs and optionally by name, where
the name is a string with no syntax, and may represent a filename,
e.g. logo.gif or a name and path within a hierarchy,
e.g. images/logos/logo1.gif

It is a goal that a Cabinet be representable as a jar file and vice
versa. A Cabinet may be converted to a jar file using a method.

One special type of Cabinet is used to store URLs, the implementation
is built around java.net.URL. The name of the ByteStore is a URL. The
cabinet is read-only, doesn’t support quotas or versioning.

ByteStores and Cabinets are added to Cabinets. Cabinets are created
in CabinetFactories and ByteStores are created in Cabinets.
*/

public interface Cabinet
    extends java.io.Serializable {

    /**
     * getProperties
     * return properties of this cabinet
     *
     * @return java.util.Map
     */
    public java.util.Map getProperties();

    /**
     * createTempByteStore
     * create new ByteStore
     * @return created ByteStore
     */
    public ByteStore createTempByteStore();

    /**
     * createByteStore
     * anonymous, known only by GUID
     * @return ByteStore
     */
    public ByteStore createByteStore();

    /**
     * createByteStore
     * named, error if name already exists
     * @param name
     * @return ByteStore
     */
    public ByteStore createByteStore(String name);

    /**
     * createByteStore
     * create new anonymous ByteStore in this Cabinet by copying contents and
     * appropriate attributes of another ByteStore.
     * @param old
     * @return ByteStore
     */
    public ByteStore createByteStore(ByteStore old);

    /**
     * createByteStore
     * create new named ByteStore in this Cabinet by copying contents and
     * appropriate attributes of another ByteStore.
     * @param name
     * @param old
     * @return ByteStore
     */
    public ByteStore createByteStore(ByteStore old);
```java
public ByteStore createByteStore(String name, ByteStore old);

/**
 * Method add
 * add subcabinet, must be from same CabinetFactory
 *
 * @param subCabinet
 *
 */
public void add(Cabinet subCabinet);

/**
 * Method add
 * add ByteStore, must be from same CabinetFactory
 *
 * @param byteStore
 *
 */
public void add(ByteStore byteStore);

/**
 * Method remove
 * remove ByteStore or Cabinet. Does not destroy Cabinet or ByteStore
 *
 * @param subCabinet
 *
 */
public void remove(Cabinet subCabinet);

/**
 * Method remove
 * remove ByteStore or Cabinet. Does not destroy Cabinet or ByteStore
 *
 * @param byteStore
 *
 */
public void remove(ByteStore byteStore);

/**
 * Method setQuota
 * set quota for Agent in this Cabinet
 *
 * @param owner
 * @param quotaBytes
 *
 */
public void setQuota(org.okiproject.serviceapi.common.Agent owner,
 long quotaBytes);

/**
 * Method getQuota
 *
 */
public long getQuota(org.okiproject.serviceapi.common.Agent owner);

/**
 * Method getQuotaUsed
 *
 */
public long getQuotaUsed(org.okiproject.serviceapi.common.Agent owner);

/**
 * Method getAvailableBytes
 *
 */
public long getAvailableBytes();

/**
 * Method getUsedBytes
 *
 */
public long getUsedBytes();

/**
 * Method getByteStore
 * get ByteStore from Cabinet by guid.
 *
 * @param guid
 *
 */
public ByteStore getByteStore(org.okiproject.serviceapi.guid.GUID guid);

/**
 * Method getByteStore
 * get by name. Not all ByteStores have names,
 * but if it has a name, it is unique within a Cabinet
 *
 * @param name
 *
 */
public ByteStore getByteStore(String name);
```

// These are analogous to jar:

/**
 * Method entries
 * analogous to jar:
 * @return java.util.Enumeration
 */
public java.util.Enumeration entries();

/**
 * Method getCabinetEntry
 * analogous to jar:
 * @param guid
 * @return CabinetEntry which has given GUID
 */
public CabinetEntry getCabinetEntry(
    org.okiproject.serviceapi.guid.GUID guid);

/**
 * Method getCabinetEntry
 * @param name
 * @return CabinetEntry which has given name
 */
public CabinetEntry getCabinetEntry(String name);

/**
 * Method returnAsJar
 * produce jar file of cabinet and its contents
 * @return java.util.jar.JarFile containing this cabinet and all its contents
 */
java.util.jar.JarFile returnAsJar();
package org.okiproject.serviceapi.file;

/*
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

/**
 * A Cabinet or a ByteStore in a Cabinet
 *
 $Source: /cvs/oki/org/okiproject/serviceapi/file/CabinetEntry.java,v $ @version $Revision: 1.6 $$Date: 2002/02/10 03:57:55 $ */

public interface CabinetEntry extends java.io.Serializable {

    /**
     * Method isCabinet
     *
     * @return boolean
     */
    public boolean isCabinet();

    /**
     * Method isByteStore
     *
     * @return boolean
     */
    public boolean isByteStore();

    /**
     * Method getParent
     *
     * @return Cabinet
     */
    public Cabinet getParent();

}
package org.okiproject.serviceapi.file;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public class CabinetException extends java.lang.Exception {
    public static final String NULLARGUMENT = "Null argument ";
    public static final String UNIMPLEMENTED = "Unimplemented Cabinet method ";
    public static final String DEFAULTCABINETFACTORYDIDNOTLOAD = "Default Roles factory did not load ";
    public static final String CABINETFACTORYPROPERTIESDIDNOTLOAD = "Roles factory properties did not load ";

    public CabinetException(String message) {
        super(message);
    }
}
package org.okiproject.serviceapi.file;

/*
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
   documentation files (the Software), to deal in the Software
   without restriction, including without limitation the rights
   to use, copy, modify, merge, publish, distribute, sublicense,
   and/or sell copies of the Software, and to permit persons
   to whom the Software is furnished to do so, subject to the
   following conditions:

   The above copyright notice and this permission notice
   shall be included in all copies or substantial portions
   of the Software.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY
   OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT
   LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
   FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT
   SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE
   FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN
   AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE
   OR OTHER DEALINGS IN THE SOFTWARE.
*/

/**
   An installation may have several implementations of CabinetFactory
   available.

   The application calls either create a CabinetFactory object and
   then invokes the methods on the object, or it looks at the
   system properties to determine which implementation to use.

   Once there is a CabinetFactory implementation available, a new Cabinet
   is created from the factory. Each implementation also provides a
   default CabinetFactory.

   Details of the implementation are application specific, e.g.,
   size of the cache, encryption algorithm, etc.

   Source: /okiproject/serviceapi/file/CabinetFactory.java, \$Revision: 1.8 $\$Date: 2002/02/10 04:38:15 $*/

public class CabinetFactory implements java.io.Serializable {

    public static CabinetFactory getDefaultCabinetFactory(
        org.okiproject.serviceapi.common.Agent agent) throws Exception {
        java.util.Map properties = getPropertiesFromPRB();
        return getCabinetFactory(agent, properties);
    }

    protected static java.util.Properties getPropertiesFromPRB() throws Exception {
        java.util.PropertyResourceBundle prb =
            (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle(                  "CabinetFactory"
        );

        if (null != prb) {
            java.util.Properties properties = new java.util.Properties();

            java.util.Enumeration keys = prb.getKeys();
            while (keys.hasMoreElements()) {
                String key = (String) keys.nextElement();
                properties.setProperty(key, (String) prb.handleGetObject(key));
            }
            return properties;
        }
        throw new org.okiproject.serviceapi.file.CabinetException(                  org.okiproject.serviceapi.file.CabinetException.CABINETFACTORYPROPERTIESDIDNOTLOAD);  }

    public static java.util.List getAvailableCabinetFactories(
        org.okiproject.serviceapi.common.Agent agent) throws Exception {
        throw new java.lang.UnsupportedOperationException();
    }

    public static CabinetFactory getCabinetFactory(
        org.okiproject.serviceapi.common.Agent agent, java.util.Properties properties) throws Exception {
        return new org.okiproject.serviceapi.file.CabinetFactory(agent, properties);  }
}

/**
   Method getAvailableCabinetFactories
   return types of CabinetFactory implementations available in the
   installation
   throw new java.lang.UnsupportedOperationException();
*/
/**
 * @param guid
 * @return Cabinet with given GUID
 */
public static Cabinet getCabinet(org.okiproject.serviceapi.guid.GUID guid) {
    throw new java.lang.UnsupportedOperationException();
}
/**
 * Method createCabinet
 * @return Cabinet
 */
public Cabinet createCabinet() {
    throw new java.lang.UnsupportedOperationException();
}
*/
private org.okiproject.serviceapi.common.Agent agent = null;
/**
 * Method getAgent
 * @return agent for which this cabinet was created
 */
public org.okiproject.serviceapi.common.Agent getAgent() {
    return this.agent;
}
/**
 * Method setAgent
 * @param agent
 */
public void setAgent(org.okiproject.serviceapi.common.Agent agent) {
    this.agent = agent;
}
private java.util.Map properties = null;
/**
 * Method getProperties
 * capability properties include:
 * replication (yes/no)
 * versioning (yes/no)
 * exclusive control to OKI system (yes/no)
 * supports quota (yes/no)
 * readonly/readwrite
 * space limit
 * degree of replication
 * replication status
 * max number of versions
 * space available
 * An application to choose an implementation from the available
 * CabinetFactory implementations which has the desired capabilities.
 * @return java.util.Map
 */
public java.util.Map getProperties() {
    return this.properties;
}
/**
 * Method getAgent
 */
/*
public static CabinetFactory getCabinetFactory(
    org.okiproject.serviceapi.common.Agent agent,
    java.util.Map properties)
throws Exception {
    throw new java.lang.UnsupportedOperationException();
}
*/
private org.okiproject.serviceapi.common.Agent agent = null;
/**
 * Method getAgent
 * @return agent for which this cabinet was created
 */
public org.okiproject.serviceapi.common.Agent getAgent() {
    return this.agent;
}
/**
 * Method setAgent
 * @param agent
 */
public void setAgent(org.okiproject.serviceapi.common.Agent agent) {
    this.agent = agent;
}
private java.util.Map properties = null;
/**
 * Method getProperties
 * capability properties include:
 * replication (yes/no)
 * versioning (yes/no)
 * exclusive control to OKI system (yes/no)
 * supports quota (yes/no)
 * readonly/readwrite
 * space limit
 * degree of replication
 * replication status
 * max number of versions
 * space available
 * An application to choose an implementation from the available
 * CabinetFactory implementations which has the desired capabilities.
 * @return java.util.Map
 */
public java.util.Map getProperties() {
    return this.properties;
}
/**
 * Method getAgent
 */
package org.okiproject.serviceapi.guid;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
<p>
Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
<p>
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
<p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
<p>
$/Source: /cvs/oki/org/okiproject/serviceapi/guid/GUID.java,v $
$version: $Revision: 1.11 $
$Date: 2002/02/08 14:25:46 $

******************************************************************************
*       Purpose: Key Abstraction representing an entity that generates GUIDs.*       Maybe this cshould be called a GUIDFactory instead.******************************************************************************
*
******************************************************************************
*/

public interface GUID
    extends java.io.Serializable {
    /**
     *    Returns an opaque GUID Object.
     * @param agent Represents the Agent state for authn/authz purposes.
     * @param type Represents the type of guid to be generated.
     * @return an opaque GUID Object.
    *
    @throw GUIDException
    */
   public Object getGUID(final org.okiproject.serviceapi.common.Agent agent,
                       final String type)
    throws GUIDException;
    //String[] getTypes(final org.okiproject.serviceapi.common.Agent agent);

    /**
     *    Admin interface to add a new GUID type.
     * @param a
     * @param type Represents the type of guid to be added.
     * @throw GUIDException
     */
    }
package org.okiproject.serviceapi.guid;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public class GUIDException
    extends java.lang.Exception {

    public static final String UNIMPLEMENTED_ERROR = "Not implemented.";
    public static final GUIDException UNIMPLEMENTED_EXCEPTION = new GUIDException(UNIMPLEMENTED_ERROR);
    public static final String DEFAULT_FACTORY_LOADING_ERROR = "Default factory did not load.";
    public static final GUIDException DEFAULT_FACTORY_LOADING_EXCEPTION = new GUIDException(DEFAULT_FACTORY_LOADING_ERROR);
    public static final String PROPERTIES_LOADING_ERROR = "Property file did not load.";
    public static final GUIDException PROPERTIES_LOADING_EXCEPTION = new GUIDException(PROPERTIES_LOADING_ERROR);

    /**
     * Constructor GUIDException
     *
     *
     */
    public GUIDException() {
        super();
    }

    /**
     * Constructor GUIDException
     *
     * @param message
     *
     */
    public GUIDException(String message) {
        super(message);
    }

package org.okiproject.serviceapi.guid;

public class GUIDFactory implements java.io.Serializable {

    static final org.okiproject.serviceapi.guid.GUIDException UNIMPLEMENTED = org.okiproject.serviceapi.guid.GUIDException.UNIMPLEMENTED_EXCEPTION;
    static final String GUID_FACTORY_NAME = "GUIDFactory";
    private java.util.Map properties;
    private org.okiproject.serviceapi.common.Agent agent;
    /** Creates a GUIDFactory with the specified properties.
     * @param a Represents the Agent state for authn/authz purposes.
     * @param props The property hash.
     * @throws Exception */
    public GUIDFactory getFactory(org.okiproject.serviceapi.common.Agent a, java.util.Properties props) throws Exception {
        java.util.Properties p = props;
        if (p == null) {
            p = getPropertiesFromPRB(GUID_FACTORY_NAME);
        }
        if (p != null) {
            throw org.okiproject.serviceapi.guid.GUIDException.PROPERTIES_LOADING_EXCEPTION;
        }
        private GUIDFactory loadFactory(final String factoryClassName, final org.okiproject.serviceapi.common.Agent agent, final java.util.Properties props)
            throws Exception {
                Class factory = Class.forName(factoryClassName);
                GUIDFactory f = (org.okiproject.serviceapi.guid.GUIDFactory) factory.newInstance();
                f.setAgent(agent);
                f.setProperties(props);
                return f;
            }
    /** Creates a GUIDFactory with the specified properties.
     * @param a Represents the Agent state for authn/authz purposes.
     * @param GUIDtype Represents the guidType required.
     * @param configFileName The config file, whether XML, property file, or
     * PropertyResourceBundle.
     * @throws Exception */
    public GUIDFactory create(org.okiproject.serviceapi.common.Agent a, String GUIDtype, String configFileName) throws Exception {
        java.io.FileInputStream fis = new java.io.FileInputStream(configFileName);
        java.util.Properties p = new java.util.Properties();
        p.load(fis);
        p.setProperty("org.okiproject.serviceapi.guid.GUIDtype", GUIDtype);
        return getFactory(a, p);
    }
    /** Creates a GUIDFactory with the specified properties.
     * @param a Represents the Agent state for authn/authz purposes.
     * @param GUIDtype Represents the guidType required.
     * @param prb The property file, whether XML, property file, or
     * PropertyResourceBundle.
     * @throws Exception */
    public GUIDFactory create(org.okiproject.serviceapi.common.Agent a, String GUIDtype, java.util.PropertyResourceBundle prb) throws Exception {
        java.util.Properties p = PRBToProperties(prb);
        p.setProperty("org.okiproject.serviceapi.guid.GUIDtype", GUIDtype);
        return getFactory(a, p);
    }
    /** Method create */
    /** Method createDefaultGUID */
}

```java
* public GUID createDefaultGUID(org.okiproject.serviceapi.common.Agent a,
  String GUIDtype, java.util.Properties props)
  throws Exception {
    props.setProperty("org.okiproject.serviceapi.guid.GUID type", GUIDtype );
    return getFactory(a, props ).getGUID( GUIDtype );
}

/** *
* Method createDefaultGUID *
* *
* @param a
* @param GUIDtype
* @param prb
* @throws Exception
* */
public GUID createDefaultGUID(org.okiproject.serviceapi.common.Agent a,
  String GUIDtype,
  java.util.PropertyResourceBundle prb)
  throws Exception {
    java.util.Properties p = PRBToProperties{ prb };
    return createDefaultGUID( a, GUIDtype, p);
}

/** *
* Method getGUID *
* *
* @param prefix -- String prefix for GUID
* @throws Exception *
* */
public GUID getGUID( String GUIDtype ) throws org.okiproject.serviceapi.guid.GUIDException
  throws Exception {
    throw UNIMPLEMENTED;
}

private static final java.util.Properties PRBToProperties{ final java.util.PropertyResourceBundle prb } throws Exception {
  if ( prb == null ) { throw org.okiproject.serviceapi.guid.GUIDException.PROPERTIES_LOADING_EXCEPTION;
    java.util.Properties properties = new java.util.Properties();
    java.util.Enumeration keys = prb.getKeys();
    while (keys.hasMoreElements()) {
      String key = (String) keys.nextElement();
      properties.setProperty(key, (String) prb.handleGetObject(key));
    }
    return properties;
  }
  return prbToProperties( prb );
}

protected void setProperties(java.util.Map properties) {
  this.properties = properties;
}
protected void setAgent(org.okiproject.serviceapi.common.Agent agent) {
  this.agent = agent;
}
```
package org.okiproject.serviceapi.guid;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public class NoMoreGUIDSException extends GUIDException {
  /**
   * Constructor NoMoreGUIDSException
   */
  public NoMoreGUIDSException() {
    super();
  }

  /**
   * Constructor NoMoreGUIDSException
   * @param message
   */
  public NoMoreGUIDSException(String message) {
    super(message);
  }
}
package org.okiproject.serviceapi.logging;

/**
 * Purpose: Represents the formatting strategy used for the message.
 *
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of
the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
 *
 * $Source: /cvs/oki/org-okiproject/serviceapi/logging/Formatter.java,v $
 * $version $Revision: 1.3 $ 
 * $Date: 2002/02/08 14:25:46 $
 *
 * Purpose: Represents the formatting strategy used for the message.
 */

public interface Formatter
extends java.io.Serializable {

/**
 * Formats input String in format specified
 *
 * @param who Represents the Agent state for authn/authz purposes.
 * @param message Input message, possibly with formatting directives.
 *
 * @return Formatted string.
 *
 */
String format(final org.okiproject.serviceapi.common.Agent who,
        final String message);
}
```java
package org.okiproject.serviceapi.logging;

/**
 * Purpose: Represents the logging service abstraction. This will typically
 * be a singleton per JVM.
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute,
 * sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface LogService
        extends java.io.Serializable {
    // Add a logger to the chain. Each logger serves as a kind of handler
    // for logging events.
    // @param who Represents the Agent state for authn/authz purposes.
    // @param logger Represents the specific logging event handler.
    // @throws LogServiceException if some error occurs while chaining
    // this handler.
    public void chainLogger(    final Logger logger) throws LogServiceException;

    // Removes a logger from the chain.
    // @param who Represents the Agent state for authn/authz purposes.
    // @param logger Represents the specific logging event handler.
    // @throws LogServiceException if some error occurs while unchaining
    // this handler.
    public void unchainLogger(    final Logger logger) throws LogServiceException;

    // Sets the threshold priority for the logging service.
    // Messages at a priority lower than this level will be ignored.
    // @param who Represents the Agent state for authn/authz purposes.
    // @param priority Represents the system logging priority.
    public void setBasePriority(    final Priority priority);

    // Gets the threshold priority for the logging service.
    // @param who Represents the Agent state for authn/authz purposes.
    public Priority getBasePriority(    final Agent who);
}
```
package org.okiproject.serviceapi.logging;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/serviceapi/logging/LogServiceException.java,v $
$Revision: 1.6 $$Date: 2002/02/08 20:33:52 $

public class LogServiceException
extends java.lang.Exception {

    public static final String UNIMPLEMENTED_ERROR = "Not implemented.";
    public static final String DEFAULT_FACTORY_LOADING_ERROR = "Default factory did not load.";
    public static final String PROPERTIES_LOADING_ERROR = "Property file did not load.";
    public static final LogServiceException UNIMPLEMENTED_EXCEPTION = new LogServiceException(UNIMPLEMENTED_ERROR);
    public static final LogServiceException DEFAULT_FACTORY_LOADING_EXCEPTION = new LogServiceException(DEFAULT_FACTORY_LOADING_ERROR);
    public static final LogServiceException PROPERTIES_LOADING_EXCEPTION = new LogServiceException(PROPERTIES_LOADING_ERROR);

    /**
     * Constructor LogServiceException
     *
     *
     */
    public LogServiceException() {
        super();
    }

    /**
     * Constructor LogServiceException
     *
     *
     * @param message
     */
    public LogServiceException(String message) {
        super(message);
    }
package org.okiproject.serviceapi.logging;

/ * * Copyright (c) 2001 Massachusetts Institute of Technology *
* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
* *
* The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
* *
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. */

public class LogServiceFactory implements java.io.Serializable {

private java.util.Map properties = null;
private org.okiproject.serviceapi.common.Agent agent;

protected static final org.okiproject.serviceapi.logging.LogServiceException UNIMPLEMENTED = org.okiproject.serviceapi.logging.LogServiceException.UNIMPLEMENTED_EXCEPTION;

/** *
* @param props *
* @throws Exception *
* */

    java.util.Map properties = props;
    if (null == properties) {
        properties = getPropertiesFromPRB();
    }
    if (null != properties) {
        String factoryClassName = (String) properties.get("LogServiceFactory");
        if (null != factoryClassName) {
            Class factoryClass = Class.forName(factoryClassName);
            LogServiceFactory factory = (LogServiceFactory) factoryClass.newInstance();
            factory.setProperties(properties);
            factory.setAgent(agent);
            return factory;
        }
    }
    throw org.okiproject.serviceapi.logging.LogServiceException.DEFAULT_FACTORY_LOADING_EXCEPTION;
}

protected static java.util.Properties getPropertiesFromPRB() throws Exception {
    java.util.PropertyResourceBundle prb = (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("LogServiceFactory");
    if (null != prb) {
        java.util.Properties properties = new java.util.Properties();
        java.utilEnumeration keys = prb.getKeys();
        while (keys.hasMoreElements()) {
            String key = (String) keys.nextElement();
            properties.setProperty(key, (String) prb.handleGetObject(key));
        }
        return properties;
    }
    throw org.okiproject.serviceapi.logging.LogServiceException.PROPERTIES_LOADING_EXCEPTION;
}

public static LogServiceFactory getDefaultLogServiceFactory() throws Exception {
    return getLogServiceFactory(agent, properties);
}

/** *
* @throws Exception *
* */

public java.util.Map getProperties() {
    return this.properties;
}

/** */
/* set Properties. */

protected void setProperties(java.util.Map properties) {
    this.properties = properties;
}

protected void setAgent( org.okiproject.serviceapi.common.Agent agent ) {
    this.agent = agent;
}

protected org.okiproject.serviceapi.common.Agent getAgent() {
    return this.agent;
}
package org.okiproject.serviceapi.logging;

/**
 * Purpose: Represents exceptions thrown by the logging service factory.
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public class LogServiceFactoryException extends Exception implements java.io.Serializable {

    /**
     * Constructor LogServiceFactoryException
     *
     */

    public LogServiceFactoryException() {
        super();
    }

    /**
     * Constructor LogServiceFactoryException
     *
     * @param mesg
     */

    public LogServiceFactoryException(final String mesg) {
        super(mesg);
    }
}
package org.okiproject.serviceapi.logging;

/**
 * Purpose: Represents the logging strategy used, allowing replaceable
 * logging strategies.
 * <p>
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * <p>
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * <p>
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
 * THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
 * OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 * <p>
 * $Source: /cvs/ok/okiproject/serviceapi/logging/Logger.java,v $
 * $Revision: 1.3 $
 * $Date: 2002/02/08 14:25:46 $
 * 
 * Purpose: Represents the logging strategy used, allowing replaceable
 * logging strategies.
 * 
 * AKA: Log [Event] Handler
 * 
 */

public interface Logger extends java.io.Serializable {

    /**
     * To open the log */
    public void start(final org.okiproject.serviceapi.common.Agent who);

    /** To close the log */
    public void stop(final org.okiproject.serviceapi.common.Agent who);

    /** Workhorse used to append to the log */
    public void appendLog(final org.okiproject.serviceapi.common.Agent who,
                          final org.okiproject.serviceapi.logging.Priority priority,
                          final String message);

    /** 
     * This is needed to tell the individual ILoggers what the base priority of
     * the logging service is. 
     * @param who Represents the Agent state for authn/authz purposes. 
     * @param priority IPriority representing the logging service’s base pr
     * iority. 
     */

    /** Other convenience methods used to append to the log. Redundant? */
    public void warning(final org.okiproject.serviceapi.common.Agent who,
                        final String message);

    /** Add an informational message to the log. */
    public void info(final org.okiproject.serviceapi.common.Agent who,
                     final String message);

    /** Add a debug level message to the log. */
    public void debug(final org.okiproject.serviceapi.common.Agent who,
                      final String message);

    /** Add an error level message to the log. */
    public void error(final org.okiproject.serviceapi.common.Agent who,
                      final String message);

    /** Add a fatal message to the log. */
    public void fatal(final org.okiproject.serviceapi.common.Agent who,
                      final String message);

    /** An interface to read the logger. It is assumed that 
     * any semantics on the format on the log will be set by 
     * the application, which will also know how to parse 
     * and handle presentation aspects as it sees fit. 
     * Only a low-level access to the contents is provided via this interface. 
     */
    public int readLog(final org.okiproject.serviceapi.common.Agent who,
                       final byte[] b, final int offset, final int count)
                throws Exception;

    /** Another interface to read the logger. This would return an enumeration 
     * of all messages matching a certain query string. 
     */
    public java.util.Iterator readLog(final org.okiproject.serviceapi.common.Agent who,
                                       final String queryString)
                throws Exception;
/** JavaBean patterns */

public void setFormatter(
    final org.okiproject.serviceapi.common.Agent who,
    final org.okiproject.serviceapi.logging.Formatter formatter);

/**
 * Method getFormatter
 *
 * @param who
 */

public org.okiproject.serviceapi.logging.Formatter getFormatter(
    final org.okiproject.serviceapi.common.Agent who);

/**
 * Method getPriority
 *
 * @param who
 */

public org.okiproject.serviceapi.logging.Priority getPriority(
    final org.okiproject.serviceapi.common.Agent who);

public void setPriority(
    final org.okiproject.serviceapi.common.Agent who, final org.okiproject.serviceapi.logging.Priority priority );

*/
package org.okiproject.serviceapi.logging;

/**
 * Purpose: Represents the priority used by either the logging subsystem or the specific logger.
 * This is a typesafe enumeration of strictly typed constants, as can be seen from the private ctor.
 */

/public (c) 2001 Massachusetts Institute of Technology

/** Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 */

/**
The SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

/** Purpose: Represents the priority used by either the logging subsystem or the specific logger.
 * This is a typesafe enumeration of strictly typed constants, as can be seen from the private ctor.
 */

$Source: /cvs/oki/org/okiproject/serviceapi/logging/Priority.java,v $ $Revision: 1.4 $ $Date: 2002/02/08 14:25:46 $

public final class Priority implements java.io.Serializable {
    private int priority;

    /** Five pre-defined system priorities */
    public static final Priority warn = new Priority(1 << 0);
    public static final Priority info = new Priority(1 << 1);
    public static final Priority debug = new Priority(1 << 2);
    public static final Priority error = new Priority(1 << 3);
    public static final Priority fatal = new Priority(1 << 4);

    /** Private ctor to prevent default instantiation. */
    private Priority(final int priority) {
        this.priority = priority;
    }

    /**
     * Method greaterOrEqual
     */
    public final static boolean greaterOrEqual(final org.okiproject.serviceapi.logging.Priority a, final org.okiproject.serviceapi.logging.Priority b) {
        return a.priority >= b.priority;
    }

    public boolean equals(Object o) {
        if (!(o instanceof org.okiproject.serviceapi.logging.Priority)) return false;

        org.okiproject.serviceapi.logging.Priority p = (org.okiproject.serviceapi.logging.Priority)o;
        return p.priority == priority;
    }

    public int hashCode() {
        return priority;
    }

    public final int getPriorityValue() {
        return priority;
    }
}

if (! (o instanceof org.okiproject.serviceapi.logging.Priority)) return false;
package org.okiproject.serviceapi.rmi;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

Source: /cvs/oki/org/okiproject/serviceapi/rmi/Client.java,v $@version@ $Revision: 1.6 $$Date: 2002/02/08 14:25:46 $
*/

public interface Client {

/**
 * This method gets the Host that this Client uses.
 */
String getHost();

/**
 * This method gets the Port that this Client uses.
 */
Integer getPort();

/**
 * This method gets the Service that this Client uses.
 */
String getService();

/**
 * This method gets the Agent for whom the Client acts.
 */
org.okiproject.serviceapi.common.Agent getForAgent();

/**
 * Given an object, a methodname, an array of method argument types,
 * and an array of arguments, this method will invoke the method.
 */
Object invoke(java.io.Serializable object, String methodName,                Class[] argTypes, Object[] args)
throws Exception;

/**
 * The test method sends a test object to the Server which returns
 * a string that indicates that the test was successful or throws
 * an exception.
 */

/**
 * @return the result of a successful test
 */
String test()
throws Exception;
}
package org.okiproject.serviceapi.rmi;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public class ClientFactory implements java.io.Serializable {

    protected static final org.okiproject.serviceapi.rmi.RmiException UNIMPLEMENTED = new org.okiproject.serviceapi.rmi.RmiException(org.okiproject.serviceapi.rmi.RmiException.UNIMPLEMENTED);

    public static org.okiproject.serviceapi.rmi.ClientFactory getDefaultClientFactory(
            org.okiproject.serviceapi.common.Agent agent, java.util.Map props)
            throws Exception {
        java.util.Map properties = props;
        if (null == properties) {
            properties = getPropertiesFromPRB();
            }
        if (null == properties) {
            String factoryClassName = (String) properties.get("ClientFactory");
            if (null != factoryClassName) {
                Class factoryClass = Class.forName(factoryClassName);
                org.okiproject.serviceapi.rmi.ClientFactory factory = (org.okiproject.serviceapi.rmi.ClientFactory) factoryClass.newInstance();
                factory.setAgent(agent);
                factory.setProperties(properties);
                return factory;
            }
            throw new org.okiproject.serviceapi.rmi.RmiException(
                    org.okiproject.serviceapi.rmi.RmiException.DEFAULTCLIENTFACTORYDIDNOTLOAD);
            }
        protected static java.util.Properties getPropertiesFromPRB()
            throws Exception {
        java.util.PropertyResourceBundle prb = (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("ClientFactory");
        if (null != prb) {
            java.util.Properties properties = new java.util.Properties();
            java.util Enumeration keys = prb.getKeys();
            while (keys.hasMoreElements()) {
                String key = (String) keys.nextElement();
                properties.setProperty(key, (String) prb.handleGetObject(key));
            }
            return properties;
        }
        throw new org.okiproject.serviceapi.rmi.RmiException(
                org.okiproject.serviceapi.rmi.RmiException.DEFAULTCLIENTFACTORYDIDNOTLOAD);
        }

    private org.okiproject.serviceapi.common.Agent agent = null;

    */

    /**
     * Method getClientFactory
     *
     * @param agent
     * @param props
     * @return org.okiproject.serviceapi.rmi.ClientFactory
     * @throws Exception
     */

    public static org.okiproject.serviceapi.rmi.ClientFactory getClientFactory(
            org.okiproject.serviceapi.common.Agent agent, java.util.Map props)
            throws Exception {
        java.util.Map properties = getPropertiesFromPRB();
        if (null == properties) {
            properties = getPropertiesFromPRB();
            }
        if (null == properties) {
            String factoryClassName = (String) properties.get("ClientFactory");
            if (null != factoryClassName) {
                Class factoryClass = Class.forName(factoryClassName);
                org.okiproject.serviceapi.rmi.ClientFactoryFactory factory = (org.okiproject.serviceapi.rmi.ClientFactoryFactory) factoryClass.newInstance();
                factory.setAgent(agent);
                factory.setProperties(properties);
                return factory;
            }
            throw new org.okiproject.serviceapi.rmi.RmiException(
                    org.okiproject.serviceapi.rmi.RmiException.DEFAULTCLIENTFACTORYDIDNOTLOAD);
            }
        }
/**
 * Method getAgent
 *
 * @return org.okiproject.serviceapi.common.Agent
 */
public org.okiproject.serviceapi.common.Agent getAgent() {
    return this.agent;
}

protected void setAgent(org.okiproject.serviceapi.common.Agent agent) {
    this.agent = agent;
}

private java.util.Map properties = null;

/**
 * Method getProperties
 *
 * @return java.util.Map
 */
public java.util.Map getProperties() {
    return this.properties;
}

protected void setProperties(java.util.Map properties) {
    this.properties = properties;
}

/**
 * Method createClient
 *
 * @param host
 * @param port
 * @param service
 * @param server
 * @param forUser
 * @throws Throwable
 */
public org.okiproject.serviceapi.rmi.Client createClient(String host,
        Integer port, String service, String server,
        org.okiproject.serviceapi.common.Agent forUser)
    throws Throwable {
    throw UNIMPLEMENTED;
}
}
package org.okiproject.serviceapi.rmi;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public class RmiException extends java.lang.Exception {
    public static final String NULLARGUMENT = "Null argument ";
    public static final String UNIMPLEMENTED = "Unimplemented Rmi method ";
    public static final String DEFAULTCLIENTFACTORYDIDNOTLOAD = "Default Client factory did not load ";
    public static final String DEFAULTSERVERFACTORYDIDNOTLOAD = "Default Server factory did not load ";
    public static final String CLIENTFACTORYPROPERTIESDIDNOTLOAD = "Client factory properties did not load ";
    public static final String SERVERFACTORYPROPERTIESDIDNOTLOAD = "Server factory properties did not load ";

    /**
     * Constructor RmiException
     * @param message
     */
    public RmiException(String message) {
        super(message);
    }
}
package org.okiproject.serviceapi.rmi;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public class ServerFactory implements java.io.Serializable {

protected static final org.okiproject.serviceapi.rmi.RmiException UNIMPLEMENTED =
new org.okiproject.serviceapi.rmi.RmiException(
org.okiproject.serviceapi.rmi.RmiException.UNIMPLEMENTED);

/**
 * Method getDefaultServerFactory
 * @param agent
 * @param props
 * @return org.okiproject.serviceapi.rmi.ServerFactory
 * @throws Exception
 */

public static org.okiproject.serviceapi.rmi.ServerFactory getDefaultServerFactory(
org.okiproject.serviceapi.common.Agent agent, java.util.Map props)
throws Exception {
java.util.Map properties = getPropertiesFromPRB();
if (null == properties) {
properties = getPropertiesFromPRB();
}
if (null != properties) {
String factoryClassName = (String) properties.get("ServerFactory");
if (null != factoryClassName) {
Class factoryClass = Class.forName(factoryClassName);
org.okiproject.serviceapi.rmi.ServerFactory factory =
(factoryClass.newInstance());
factory.setAgent(agent);
factory.setProperties(properties);
factory.setup(properties);
return factory;
}
throw new org.okiproject.serviceapi.rmi.RmiException(
org.okiproject.serviceapi.rmi.RmiException.DEFAULTSERVERFACTORYDIDNOTLOAD);
}

protected static java.util.Properties getPropertiesFromPRB() throws Exception {
java.util.PropertyResourceBundle prb =
(java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("ServerFactory");
if (null != prb) {
java.util.Properties properties = new java.util.Properties();
java.utilEnumeration keys = prb.getKeys();
while (keys.hasMoreElements()) {
String key = (String) keys.nextElement();
properties.setProperty(key, (String) prb.handleGetObject(key));
}
return properties;
}
throw new org.okiproject.serviceapi.rmi.RmiException(
org.okiproject.serviceapi.rmi.RmiException.SERVERFACTORYPROPERTIESDIDNOTLOAD);
}

private org.okiproject.serviceapi.common.Agent agent = null;
/**
 * Method getAgent
 */

public static org.okiproject.serviceapi.rmi.ServerFactory getServerFactory(
org.okiproject.serviceapi.common.Agent agent, java.util.Map props)
org.okiproject.serviceapi.rmi/ServerFactory.java

    *   *   @return org.okiproject.serviceapi.common.Agent   *   */
    public org.okiproject.serviceapi.common.Agent getAgent() {
        return this.agent;
    }

    protected void setAgent(org.okiproject.serviceapi.common.Agent agent) {
        this.agent = agent;
    }

    private java.util.Map properties = null;

    /**
     * Method getProperties
     *   *   @return java.util.Map
     *   */
    public java.util.Map getProperties() {
        return this.properties;
    }

    protected void setProperties(java.util.Map properties) {
        this.properties = properties;
    }

    /**
     * Method setup
     *   *   @param props
     *   *   @throws Exception
     *   */
    public void setup(java.util.Map props) throws Exception {
        throw UNIMPLEMENTED;
    }
}
package org.okiproject.serviceapi.roles;

/**
  Copyright (c) 2001 Massachusetts Institute of Technology
  <p>Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
</p>
<p>The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
</p><p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
</p>
<p>$Source: /cvs/oki/org/okiproject/serviceapi/roles/Authorization.java,v $
$Revision: 1.6 $ $Date: 2002/02/08 14:25:46 $
*/

public interface Authorization extends java.io.Serializable {

/**
  Method getCanForUserGrant
  */
  org.okiproject.serviceapi.roles.GrantPrivilege getCanForUserGrant();

/**
  Method getDoFunction
  */
  Boolean getDoFunction();

/**
  Method getEffectiveDate
  */
  java.sql.Timestamp getEffectiveDate();

/**
  Method getExpirationDate
  */
  java.sql.Timestamp getExpirationDate();

/**
  Method getFunction
  */
  org.okiproject.serviceapi.roles.Function getFunction();

/**
  Method getGrant
  */
  org.okiproject.serviceapi.roles.GrantPrivilege getGrant();

/**
  Method getId
  */
  Object getId();

/**
  Method getModifiedBy
  */
  String getModifiedBy();

/**
  Method getModifiedDate
  */
  java.sql.Timestamp getModifiedDate();

/**
  Method getQualifier
  */
  org.okiproject.serviceapi.roles.Qualifier getQualifier();

/**
  Method getPerson
  */
  org.okiproject.serviceapi.roles.Person getPerson();

/**
  Method isActiveNow
  */
  Boolean isActiveNow();
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/serviceapi/roles/Authorizations.java,v $
$Revision: 1.3 $
$Date: 2002/02/08 14:25:46 $

* @see org.okiproject.serviceapi.roles.Authorization
* @see org.okiproject.serviceapi.roles.RolesFactory
*/

public interface Authorizations
extends java.io.Serializable {

/**
 * Method hasMoreAuthorizations
 * @throws Exception
 */

boolean hasMoreAuthorizations() throws Exception;

/**
 * Method nextAuthorization
 * @throws Exception
 */

org.okiproject.serviceapi.roles.Authorization nextAuthorization() throws Exception;
}
package org.okiproject.serviceapi.roles;

/**<p>
Copyright (c) 2001 Massachusetts Institute of Technology
</p><p>
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
</p><p>
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
</p><p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
</p>*/
Source: /cvs/oki/org/okiproject/serviceapi/roles/Field.java,v $Revision: 1.4 $$Date: 2002/02/08 14:25:46$
* @see org.okiproject.serviceapi.roles.FieldValue
* @see org.okiproject.serviceapi.roles.RolesFactory
*
public class Field
implements java.io.Serializable {

private Object id = null;

/**
 * Method getId
 */
public Object getId() {
    return this.id;
}

/**
 * Method setId
 */
public void setId(Object id) {
    this.id = id;
}

private String name = null;

/**
 * Method getName
 */
public String getName() {
    return this.name;
}

/**
 * Method setName
 */
public void setName(String name) {
    this.name = name;
}

private String description = null;

/**
 * Method getDescription
 */
public String getDescription() {
    return this.description;
}

/**
 * Method setDescription
 */
public void setDescription(String description) {
    this.description = description;
}

/**
 * Constructor Field
 */
public Field(String name) {
    this.name = name;
}

/**
 * Constructor Field
 */
public Field(String name, String description) {
    this.name = name;
    this.description = description;
}
/** Constructor Field */

/** @param id */

public Field(Object id) {
  
  this.id = id;
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
public class FieldValue implements java.io.Serializable {
    private org.okiproject.serviceapi.roles.Field field = null;

    /**
     * Method getField
     */
    public org.okiproject.serviceapi.roles.Field getField() {
        return this.field;
    }

    /**
     * Method setValue
     * @param value
     */
    public void setValue(Object value) {
        this.value = value;
    }

    private Object value = null;
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public interface Function extends java.io.Serializable {

    /** *
     * Method getDescription *
     */
    String getDescription();

    /** *
     * Method getFunctionCategory *
     */
    org.okiproject.serviceapi.roles.FunctionCategory getFunctionCategory();

    /** *
     * Method getId *
     */
    Object getId();

    /** *
     * Method getModifiedBy *
     */
    String getModifiedBy();

    /** *
     * Method getModifiedDate *
     */
    java.sql.Timestamp getModifiedDate();

    /** *
     * Method getName *
     */
    String getName();

    /** *
     * Method getProperties *
     */
    java.util.Properties getProperties();

    /** *
     * Method getQualifierType *
     */
    org.okiproject.serviceapi.roles.QualifierType getQualifierType();

    /** *
     * Method setFunctionCategory *
     */
    void setFunctionCategory(
        org.okiproject.serviceapi.roles.FunctionCategory functionCategory)
        throws Exception;

    /** *
     * Method setId *
     */
    void setId(Object id);

    /** *
     * Method setName *
     */
    void setName(String name);

    /** *
     * Method setQualifierType *
     */
}
void setQualifierType(
    org.okiproject.serviceapi.roles.QualifierType qualifierType)
    throws Exception;

package org.okiproject.serviceapi.roles;

/**
 */

Copyright (c) 2001 Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

public interface FunctionCategory
extends java.io.Serializable {

/**
 * Method getCategory
 *
 */
String getCategory();

/**
 * Method getDescription
 *
 */
String getDescription();

/**
 * Method setCategory
 *
 */
void setCategory(String category);
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
<p>
 Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
</p>
<p>
 The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
</p>
<p>
 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
</p>

public interface Functions
extends java.io.Serializable {

    /**
     * Method hasMoreFunctions
     * @throws Exception
     */
    boolean hasMoreFunctions() throws Exception;

    /**
     * Method nextFunction
     * @throws Exception
     */
    org.okiproject.serviceapi.roles.Function nextFunction() throws Exception;
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

// Permission is hereby granted, free of charge, to any
// person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

/**
 * The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
 */

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

/**
 $Source: /cvs/oki/org/okiproject/serviceapi/roles/GrantPrivilege.java,v $
 $Revision: 1.3 $
 $Date: 2002/02/08 14:25:46 $
 */

public interface GrantPrivilege
extends java.io.Serializable {

// final static String NOTGRANTABLE = "N";
public final static String NOTGRANTABLE = "N";

// final static String GRANTABLE = "G";
public final static String GRANTABLE = "G";

// final static String GRANTABLEFORGRANTING = "GG";
public final static String GRANTABLEFORGRANTING = "GG";

/**
 * Method isGrantable
 */
public boolean isGrantable();

/**
 * Method isGrantableForGranting
 */
public boolean isGrantableForGranting();

/**
 * Method toString
 */
String toString();
package org.okiproject.serviceapi.roles;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute,
sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/serviceapi/roles/Person.java,v $
$@version$ $Revision: 1.5 $ $Date: 2002/02/08 14:25:46 $
* @see org.okiproject.serviceapi.roles.Field
* @see org.okiproject.serviceapi.roles.RolesFactory
*/
public interface Person
extends org.okiproject.serviceapi.common.Person {

/**
   * Determines if this Person has a matching Person in the Roles DB.
   *
   * @return true indicates Person is in the Roles DB as are its properties.
   * @return false indicates Roles DB hasn’t been queried or Person is not in t
   *        he Roles DB.
   */
   boolean isFoundInRoles()
   throws Exception;

/**
   * Refresh properties from a changing Roles DB.
   */
   void refresh()
   throws Exception;
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public interface Qualifier extends java.io.Serializable {

  /**
   * Method getId
   */
  Object getId();

  /**
   * Method getCode
   */
  String getCode();

  /**
   * Method getDescription
   */
  String getDescription();

  /**
   * Method getHasChild
   */
  Boolean getHasChild();

  /**
   * Method getQualifierType
   */
  org.okiproject.serviceapi.roles.QualifierType getQualifierType();

  /**
   * Method setId
   */
  void setId(Object id);

  /**
   * Method setCode
   */
  void setCode(String code);

  /**
   * Method setQualifierType
   */
  void setQualifierType(org.okiproject.serviceapi.roles.QualifierType qualifierType);

  /**
   * Adds a parent-child relationship between two existing qualifiers in the Roles DB.
   */
  void addLink(org.okiproject.serviceapi.roles.Qualifier parent)
      throws Exception;

  /**
   * Deletes a parent-child relationship between two existing qualifiers in the Roles DB.
   */
  void deleteLink()
      throws Exception;

  /**
   * For an existing qualifier, moves the parent link from one existing qualifier to another.
   */
  void moveLink(org.okiproject.serviceapi.roles.Qualifier oldParent,
                org.okiproject.serviceapi.roles.Qualifier newParent);

  /**
   * Method getProperties
   */
  java.util.Properties getProperties();

  /**
   * Method getQualifierType
   */
  org.okiproject.serviceapi.roles.QualifierType getQualifierType();

  /**
   * Method setId
   */
  void setId(Object id);

  /**
   * Method setCode
   */
  void setCode(String code);

  /**
   * Method setQualifierType
   */
  void setQualifierType(org.okiproject.serviceapi.roles.QualifierType qualifierType);

  /**
   * Adds a parent-child relationship between two existing qualifiers in the Roles DB.
   */
  void addLink(org.okiproject.serviceapi.roles.Qualifier parent)
      throws Exception;

  /**
   * Deletes a parent-child relationship between two existing qualifiers in the Roles DB.
   */
  void deleteLink()
      throws Exception;

  /**
   * For an existing qualifier, moves the parent link from one existing qualifier to another.
   */
  void moveLink(org.okiproject.serviceapi.roles.Qualifier oldParent,
                org.okiproject.serviceapi.roles.Qualifier newParent);
org/okiproject/serviceapi/roles/Qualifier.java

```java
throws Exception;

/**
 * For an existing qualifier, determines if the qualifier is the child a
 * given parent
 */

//Boolean isChildOf( org.okiproject.serviceapi.roles.Qualifier parent ) throws Exception;

/**
 * For an existing qualifier, get its descendants
 */

//org.okiproject.serviceapi.roles.Qualifiers getDescendents() throws Exception;

}```
package org.okiproject.serviceapi.roles;

/**
*p>  Copyright (c) 2001 Massachusetts Institute of Technology
</p>
<p>  Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
</p>
<p>  The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
</p>
<p>  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
</p>
<p>
$Source: /cvs/oki/org/okiproject/serviceapi/roles/QualifierType.java,v $
$Revision: 1.4 $
$Date: 2002/02/08 14:25:46 $

* @see org.okiproject.serviceapi.roles.Qualifier
* @see org.okiproject.serviceapi.roles.RolesFactory
*/
public interface QualifierType
extends java.io.Serializable {

/**
 * Method getType
 * *
 */
String getType();

/**
 * Method setType
 * *
 * @param type
 * *
 */
void setType(String type);

/**
 * Method getDescription
 * *
 */
String getDescription();
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
<br>
<br>Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
</br>
<br>The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
</br>
<br>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
</br>
<br>$Source: /cvs/oki/org/okiproject/serviceapi/roles/Qualifiers.java,v $
$Revision: 1.3 $
$Date: 2002/02/08 14:25:46 $
*/

public interface Qualifiers
extends java.io.Serializable {

    /**
     * Method hasMoreQualifiers
     * @throws Exception
     */
    boolean hasMoreQualifiers() throws Exception;

    /**
     * Method nextQualifier
     * @throws Exception
     */
    org.okiproject.serviceapi.roles.Qualifier nextQualifier() throws Exception;
}
package org.okiproject.serviceapi.roles;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public class RolesException

public static final String INVALIDPERSONKEY = "Invalid person key ";
public static final String PERSONNOTFOUNDINROLES = "Person not found in Roles ";
public static final String PEOPLEENUMERATIONFAILED = "Enumeration of people failed ";
public static final String INVALIDQUALIFIER = "Invalid qualifier ";
public static final String QUALIFIERNOTFOUNDINROLES = "Qualifier not found in Roles ";
public static final String QUALIFIERTYPENOTFOUNDINROLES = "Qualifier type not found in Roles ";
public static final String QUALIFIERENUMERATIONFAILED = "Enumeration of qualifiers failed ";
public static final String INVALIDGRANTPRIVILEGE = "Invalid grant privilege ";
public static final String NULLARGUMENT = "Null argument ";
public static final String UNIMPLEMENTED = "Unimplemented Roles method ";
public static final String DEFAULTROLESFACTORYDIDNOTLOAD = "Default Roles factory did not load ";
public static final String DATABASESYMBOLICNAMELISTDIDNOTLOAD = "Database symbolic name list did not load ";
public static final String MISSINGDATABASESYMBOLICNAME = "Current database symbolic name is not set ";
public static final String ROLESFACTORYPROPERTIESDIDNOTLOAD = "Roles factory properties did not load ";

public static final String QUERYAUTHORIZATIONFAILED = "Query authorization failed ";
public static final String CREATEAUTHORIZATIONFAILED = "Create authorization failed ";
public static final String DELETEAUTHORIZATIONFAILED = "Delete authorization failed ";
public static final String UPDATEAUTHORIZATIONFAILED = "Update authorization failed ";
public static final String AUTHORIZATIONENUMERATIONFAILED = "Enumeration of authorizations failed ";
public static final String AUTHORIZATIONFIELDDOESNOTEXIST = "Authorization field does not exist ";
public static final String FUNCTIONIDALREADYEXISTS = "Function already exists ";
public static final String INVALIDDATABASE = "Invalid database ";
public static final String INVALIDFUNCTION = "Invalid function ";
public static final String FUNCTIONNOTFOUNDINROLES = "Function not found in Roles ";
public static final String FUNCTIONCATEGORYNOTFOUNDINROLES = "Function category not found in Roles ";
public static final String FUNCTIONENUMERATIONFAILED = "Enumeration of functions failed ";
public static final String INVALIDQUALIFIER = "Invalid qualifier ";
public static final String QUALIFIERNOTFOUNDINROLES = "Qualifier not found in Roles ";
public static final String QUALIFIERTYPENOTFOUNDINROLES = "Qualifier type not found in Roles ";
public static final String QUALIFIERENUMERATIONFAILED = "Enumeration of qualifiers failed ";
public static final String INVALIDPERSONKEY = "Invalid person key ";
public static final String PERSONNOTFOUNDINROLES = "Person not found in Roles ";
public static final String PEOPLEENUMERATIONFAILED = "Enumeration of people failed ";

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
*/

public class RolesException

public RolesException(String message) {
    super(message);
}
package org.okiproject.serviceapi.roles;

/** Copyright (c) 2001 Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute,
sub-license, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

*/

RolesFactory</h2>
The RolesFactory allows the application developer to create Roles
objects in the Roles database, to get Roles objects from the Roles
database given selection criteria, to ask questions of Roles such as
who can do what with whom.

The key objects in Roles are Authorizations, Functions, People, and
and Qualifiers. There are some other objects in Roles that play
supporting roles. They are FunctionCategory, QualifierType, Field,
and FieldValue.

There are two methods to create Authorizations. One uses method uses
Person, Function, and Qualifier. The other adds effective date and
expiration date. For the method without the dates the effective date
is today and there is no expiration date.

There are methods for getting Authorizations:

(1) getAuthsSimpleForm

This method is overloaded to allow for combinations
of filtering arguments to find matching Authorizations.

(2) getAuthsForDoingFWithQ

Given a Function and a Qualifier, finds a list of all
Authorizations that would permit a person to do the
Function with the Qualifier. This method is overloaded
so that Authorizations for Functions within a FunctionCategory
can be found. This uses a different algorithm than
getAuthsSimpleForm.

Note that the Qualifier in any found Authorization
might be a parent of the Qualifier give as an argument.

(3) getAuthsGeneralForm

Give a set of criteria/value pairs, this method finds
all matching Authorizations. With the general form
the application developer can do anything that the
other two forms of getAuths can do and a lot more.

*/

public static final org.okiproject.serviceapi.roles(RolesException UNIMPLEMENTED =
new org.okiproject.serviceapi.roles.RolesException(org.okiproject.serviceapi.roles.RolesException.UNIMPLEMENTED);

private static final org.okiproject.serviceapi.roles.RolesException MISSINGDATABASESYMBOLICNAME =
new org.okiproject.serviceapi.roles.RolesException(MISSINGDATABASESYMBOLICNAME);

/**
 * Method getDefaultRolesFactory
 *
 * @param agent
 * @param props
 */

public static final org.okiproject.serviceapi.roles.RolesFactory getDefaultValue

public static org.okiproject.serviceapi.roles.RolesFactory getRolesFactory(
    org.okiproject.serviceapi.common.Agent agent, java.util.Map props)
throws Exception {
    java.util.Map properties = props;
    if (null == properties) {
        properties = getPropertiesFromPRB();
    }
    if (null != properties) {
        String factoryClassName =
            (String) properties.get("RolesFactory");
        Class factoryClass =
            Class.forName(factoryClassName);
        org.okiproject.serviceapi.roles.RolesFactory factory =
            (org.okiproject.serviceapi.roles.RolesFactory) factoryClass.newInstance();
        factory.setProperties(properties);
        return factory;
    }
    throw new org.okiproject.serviceapi.roles.RolesException(
        org.okiproject.serviceapi.roles.RolesException.DEFAULTROLESFACTORYDIDNOTLOAD);
}

protected static java.util.Properties getPropertiesFromPRB()
throws Exception {
    java.util.PropertyResourceBundle prb =
        (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle(
            "RolesFactory");
    if (null != prb) {
        java.util.Properties properties = new java.util.Properties();
        java.util.Enumeration keys = prb.getKeys();
        while (keys.hasMoreElements()) {
            String key = (String) keys.nextElement();
            properties.setProperty(key, (String) prb.handleGetObject(key));
        }
        return properties;
    }
    throw new org.okiproject.serviceapi.roles.RolesException(
        org.okiproject.serviceapi.roles.RolesException.ROLESFACTORYPROPERTIESDIDNOTLOAD);
}

private org.okiproject.serviceapi.common.Agent agent = null;

/**
 * Method getAgent
 *
 * @return org.okiproject.serviceapi.common.Agent
 */
public org.okiproject.serviceapi.common.Agent getAgent() {
    return this.agent;
}

/**
 * Method setAgent
 *
 * @param agent
 */
public void setAgent(org.okiproject.serviceapi.common.Agent agent) {
    this.agent = agent;
}

private java.util.Map properties = null;

/**
 * Method getProperties
 *
 * @return java.util.Map
 */
public java.util.Map getProperties() {
    return this.properties;
}

/**
 * Method setProperties
 *
 * @param properties
 */
public void setProperties(java.util.Map properties) {
    this.properties = properties;
}

/**
 * Method getDatabaseSymbolicNames
 *
 * @return String[]
 * @throws Exception
 */
public static String[] getDatabaseSymbolicNames()
throws Exception {
    java.util.Properties properties = getPropertiesFromPRB();
    if (null != properties) {
        String countStr = properties.getProperty("CountDatabaseSymbolicNames");
        if (null != countStr) {
            final int count = Integer.parseInt(countStr);
            String[] names = new String[count];
            for (int i = 0; count != i; i++) {
                String prop = "DatabaseSymbolicName" + Integer.toString(i + 1);
                names[i] = properties.getProperty(prop);
            }
        }
    }
}
/**
 * Method getDatabaseSymbolicName
 * @return String
 * @throws Exception
 */
public String getDatabaseSymbolicName() throws Exception {
    if (null != this.properties) {
        if (this.properties.containsKey("CurrentDatabaseSymbolicName")) {
            return (String) this.properties.get("CurrentDatabaseSymbolicName");
        }
    }
    throw MISSINGDATABASESYMBOLICNAME;
}

/**
 * Method setDatabaseSymbolicName
 * @param databaseSymbolicName
 * @throws Exception
 */
public void setDatabaseSymbolicName(String databaseSymbolicName) throws Exception {
    if (null != this.properties) {
        this.properties.put("CurrentDatabaseSymbolicName", databaseSymbolicName);
    }
    throw MISSINGDATABASESYMBOLICNAME;
}

/**
 * Creates a new Authorization in the Roles DB for a Person
 * doing a Function with a Qualifier and having Grant rights
 * and ability to doFunction. Uses current date/time as the
 * effectiveDate and doesn’t set an expiration date.
 */
/**
 * Method createAuthorization
 * @param person
 * @param function
 * @param qualifier
 * @param grant
 * @param doFunction
 * @param effectiveDate
 * @param expirationDate
 * @return org.okiproject.serviceapi.roles.Authorization
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Authorization createAuthorization(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.Function function,
    ...)
    throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Deletes an existing authorization in the Roles DB.
 */
/**
 * Method deleteAuthorization
 * @param authorization
 * @throws Exception
 */
public void deleteAuthorization();
org.okiproject.serviceapi.roles/RolesFactory.java

```java
public void updateAuthorization(
    org.okiproject.serviceapi.roles.Authorization authorization,
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.Function function,
    org.okiproject.serviceapi.roles.Qualifier qualifier,
    boolean grant,
    boolean doFunction,
    java.sql.Timestamp effectiveDate,
    java.sql.Timestamp expirationDate)
    throws Exception {
    throw UNIMPLEMENTED;
}

public org.okiproject.serviceapi.common.Agents getWhoCanDo(
    org.okiproject.serviceapi.roles.Function function,
    org.okiproject.serviceapi.roles.Qualifier qualifier,
    boolean isActiveNow)
    throws Exception {
    throw UNIMPLEMENTED;
}

public org.okiproject.serviceapi.common.Agents getWhoCanDo(
    org.okiproject.serviceapi.roles.Function function,
    boolean isActiveNow)
    throws Exception {
    throw UNIMPLEMENTED;
}

public org.okiproject.serviceapi.common.Agents getWhoCanDo(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    boolean isActiveNow)
    throws Exception {
    throw UNIMPLEMENTED;
}
```

/**
 * Given a Function
 * returns an enumeration of all people at least one
 * Authorization for the Function where the Qualifier
 * is not considered.
 */

/**
 * Method getWhoCanDo
 *
 * @param functionCategory
 * @param isActiveNow
 *
 * @return org.okiproject.serviceapi.common.Agents
 * @throws Exception
 */

public org.okiproject.serviceapi.common.Agents getWhoCanDo(
    org.okiproject.serviceapi.roles.Function functionCategory,
    boolean isActiveNow)
    throws Exception {
    throw UNIMPLEMENTED;
}
public org.okiproject.serviceapi.common.Agents getWhoCanDo(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    org.okiproject.serviceapi.roles.Qualifier qualifier, 
    Boolean isActiveNow)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Given a Qualifier
 * returns an enumeration of all people with at least one
 * Authorization for at least one Function with the Qualifier.
 */

/** Method getAuthsForDoingFWithQ
 * @param functionCategory
 * @param qualifier
 * @param isActiveNow
 * @param function
 * @param qualifier
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 * @return org.okiproject.serviceapi.roles.Authorizations
 * @throws Exception
 */

public org.okiproject.serviceapi.roles.Authorizations getAuthsForDoingFWithQ(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory, 
    org.okiproject.serviceapi.roles.Qualifier qualifier, 
    Boolean isActiveNow)
throws Exception {
    throw UNIMPLEMENTED;
}
/* Given a Person, a FunctionCategory, and a Qualifier
 * returns an enumeration of matching authorizations.
 * The authorizations must be for Functions within the
 * given FunctionCategory.
 */

/**
 * Method getAuthsSimpleForm
 *
 * @param person
 * @param functionCategory
 * @param qualifier
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 *
 * @return org.okiproject.serviceapi.roles.Authorizations
 *
 * @throws Exception
 *
 */

public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    org.okiproject.serviceapi.roles.Qualifier qualifier,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
    throws Exception {
    throw UNIMPLEMENTED;
}

/* Given a Person and a Function
 * returns an enumeration of Authorizations
 * matching the Person and the Function where
 * Qualifier is not considered.
 */

/**
 * Method getAuthsSimpleForm
 *
 * @param person
 * @param function
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 *
 * @return org.okiproject.serviceapi.roles.Authorizations
 *
 * @throws Exception
 *
 */

public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.Function function,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
    throws Exception {
    throw UNIMPLEMENTED;
}

/* Given a Person and a FunctionCategory
 * returns an enumeration of Authorizations
 * matching the Person and any Function in
 * the given FunctionCategory where Qualifier
 * is not considered.
 */

/**
 * Method getAuthsSimpleForm
 *
 * @param person
 * @param functionCategory
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 *
 * @return org.okiproject.serviceapi.roles.Authorizations
 *
 * @throws Exception
 *
 */

public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
    throws Exception {
    throw UNIMPLEMENTED;
}

/* Given a Person and a Qualifier
 * returns an enumeration of Authorizations
 * matching Person and Qualifier where
 * Function is not considered.
 */

/**
 * Method getAuthsSimpleForm
 *
 * @param person
 * @param qualifier
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 *
 * @return org.okiproject.serviceapi.roles.Authorizations
 *
 * @throws Exception
 *
 */

public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.Qualifier qualifier,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
    throws Exception {
    throw UNIMPLEMENTED;
}
/* Given a Person
 * returns an enumeration of Authorizations
 * matching the Person where Function and
 * Qualifier are not considered.
 */
/**
 * Method getAuthsSimpleForm
 * @param person
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 * @return org.okiproject.serviceapi.roles.Authorizations
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Person person, Boolean isActiveNow,
    Boolean expandFunctions, Boolean expandQualifiers)
throws Exception {
    throw UNIMPLEMENTED;
}

/* Given a Function and a Qualifier
 * returns an enumeration of Authorizations
 * matching the Function and the Qualifier
 * where Person is not considered.
 */
/**
 * Method getAuthsSimpleForm
 * @param function
 * @param qualifier
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 * @return org.okiproject.serviceapi.roles.Authorizations
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Function function, org.okiproject.serviceapi.roles.Qualifier qualifier,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
throws Exception {
    throw UNIMPLEMENTED;
}

/* Given a Qualifier
 * returns an enumeration of Authorizations
 * matching the given Qualifier where Function
 * and Person are not considered.
 */
/**
 * Method getAuthsSimpleForm
 * @param qualifier
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 * @return org.okiproject.serviceapi.roles.Authorizations
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Qualifier qualifier,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
throws Exception {
    throw UNIMPLEMENTED;
}

/* Given a FunctionCategory and a Qualifier
 * returns an enumeration of Authorizations
 * matching Functions of the given FunctionCategory
 * and Qualifier where Person is not considered.
 */
/**
 * Method getAuthsSimpleForm
 * @param functionCategory
 * @param qualifier
 * @param isActiveNow
 * @param expandFunctions
 * @param expandQualifiers
 * @return org.okiproject.serviceapi.roles.Authorizations
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory, org.okiproject.serviceapi.roles.Qualifier qualifier,
    Boolean isActiveNow, Boolean expandFunctions, Boolean expandQualifiers)
throws Exception {
    throw UNIMPLEMENTED;
}
public org.okiproject.serviceapi.roles.Authorizations getAuthsSimpleForm(
    org.okiproject.serviceapi.roles.Function function,
    Boolean isActiveNow, Boolean expandFunctions,
    Boolean expandQualifiers)
throws Exception {
    throw UNIMPLEMENTED;
}

/*
* Given a list of selection criteria field/value pairs
* returns an enumeration of Authorizations.
*/

/** *
* Method getAuthsGeneralForm
* *
* @return org.okiproject.serviceapi.roles.Authorizations
* @throws Exception
*/
public org.okiproject.serviceapi.roles.Authorizations getAuthsGeneralForm(
    org.okiproject.serviceapi.roles.FieldValue[] selectionCriteria)
throws Exception {
    throw UNIMPLEMENTED;
}

/*
* @return the minimum list of Authorization fields.
*/

/** *
* Method getMinimumAuthorizationFields
* *
* @return org.okiproject.serviceapi.roles.Field[]
* @throws Exception
*/
public org.okiproject.serviceapi.roles.Field[] getMinimumAuthorizationFields()
throws Exception {
    throw UNIMPLEMENTED;
}

/*
* @return the default list of Authorization fields.
*/

/** *
* Method getDefaultAuthorizationFields
* *
* @return org.okiproject.serviceapi.roles.Field[]
* @throws Exception
*/
public org.okiproject.serviceapi.roles.Field[] getDefaultAuthorizationFields()
throws Exception {
    throw UNIMPLEMENTED;
}

/*
* @return the current list of Authorization fields.
*/

/** *
* Method getCurrentAuthorizationFields
* *
* @param currentFields
* @throws Exception
*/
public void setCurrentAuthorizationFields(
    org.okiproject.serviceapi.roles.Field[] currentFields)
throws Exception {
    throw UNIMPLEMENTED;
}
throw UNIMPLEMENTED;
}

/**
 * Given a Person, Function, and Qualifier
 * returns true if the Person is authorized to perform
 * the given Function with the given Qualifier.
 */

/**
 * Method isAuthorized
 *
 * @param person
 * @param function
 * @param qualifier
 *
 * @return Boolean
 *
 * @throws Exception
 */

public Boolean isAuthorized(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.Function function,
    org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Given a Person and Qualifier
 * returns true if the Person is authorized to perform
 * at least one Function with the given Qualifier.
 */

/**
 * Method isAuthorized
 *
 * @param person
 * @param qualifier
 *
 * @return Boolean
 *
 * @throws Exception
 */

public Boolean isAuthorized(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Given a Person, FunctionCategory, and Qualifier
 * returns true if the Person is authorized to perform
 * at least one Function in the FunctionCategory with the Qualifier.
 */

/**
 * Method isAuthorized
 *
 * @param person
 * @param functionCategory
 * @param qualifier
 *
 * @return Boolean
 *
 * @throws Exception
 */

public Boolean isAuthorized(
    org.okiproject.serviceapi.roles.Person person,
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Creates a new function in the Roles DB.
 */

public org.okiproject.serviceapi.roles.Function createFunction(String name,
    String description,
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    org.okiproject.serviceapi.roles.QualifierType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Deletes an existing function in the Roles DB.
 */

public void deleteFunction(
    org.okiproject.serviceapi.roles.Function function)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Determines if this Function has a matching Function in the Roles DB.
 */


public Boolean isFunctionFoundInRoles(OrgFunction function) throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Method refreshFunction
 *
 * @param function
 * @throws Exception
 */
public void refreshFunction(OrgFunction function) throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Updates one or more fields in an existing function in the Roles DB.
 * @see org.okiproject.serviceapi.roles.FunctionCategory
 * @see org.okiproject.serviceapi.roles.QualifierType
 */
public void updateFunction(OrgFunction function, String newName, String newDescription,
    org.okiproject.serviceapi.roles.FunctionCategory newFunctionCategory,
    org.okiproject.serviceapi.roles.QualifierType newQualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Determines if this FunctionCategory has a matching FunctionCategory in the Roles DB.
 * @param functionCategory
 * @return true indicates FunctionCategory is in the Roles DB as are its properties.
 * @return false indicates Roles DB hasn’t been queried or FunctionCategory is not in the Roles DB.
 */
public org.okiproject.serviceapi.roles.FunctionCategory createFunctionCategory(String category, String description)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Deletes an existing FunctionCategory in the Roles DB.
 */
public void deleteFunctionCategory(OrgFunctionCategory functionCategory)
throws Exception {
    throw UNIMPLEMENTED;
}
org/okiproject/serviceapi/roles/ RolesFactory.java

```java
public Boolean isFunctionCategoryFoundInRoles(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
   * Refreshs FunctionCategory properties from a changing Roles DB.
   */
public void refreshFunctionCategory(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
   * Updates description in an existing FunctionCategory in the Roles DB.
   */
public void updateFunctionCategory(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory,
    String newDescription)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
   * @return Function categories available to Roles
   */
public org.okiproject.serviceapi.roles.FunctionCategory[]
getFunctionCategories()
throws Exception {
    throw UNIMPLEMENTED;
}

/**
   * Given a FunctionCategory
   * returns an enumeration of Functions from the Roles DB
   */
public org.okiproject.serviceapi.roles.Functions
getFunctionsInCategory(
    org.okiproject.serviceapi.roles.FunctionCategory functionCategory)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
   * Method getFunctionsGeneralForm
   * Given a list of selection criteria field/value pairs
   * returns an enumeration of Functions.
   */
public org.okiproject.serviceapi.roles.Functions getFunctionsGeneralForm(
    org.okiproject.serviceapi.roles.FieldValue[] selectionCriteria)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
   * Gets the minimum list of Function fields.
   */
public org.okiproject.serviceapi.roles.Field[]
getMinimumFunctionFields()
throws Exception {
    throw UNIMPLEMENTED;
}
```

/**
 * Gets the default list of Function fields.
 */

/**
 * @return org.okiproject.serviceapi.roles.Field[]
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Field[] getDefaultFunctionFields()
    throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * @return the complete list of Function fields.
 */

/**
 * @return org.okiproject.serviceapi.roles.Field[]
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Field[] getCompleteFunctionFields()
    throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * @return the current list of Function fields.
 */

/**
 * @param currentFields
 * @throws Exception
 */
public void setCurrentFunctionFields(
    org.okiproject.serviceapi.roles.Field[] currentFields)
    throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * @param functionCategory
 * @param name
 * @return new instance of Function. The instance is local to the application.
 * It may or may not exist in Roles.
 */

/**
 * @param functionCategory
 * @param name
 * @return org.okiproject.serviceapi.roles.Function
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Function newFunction(
    String functionCategory,
    String name)
    throws Exception {
    throw UNIMPLEMENTED;
}
/**
 * Returns a new instance of FunctionCategory. The instance is local to the application.
 * It may or may not exist in the Roles DB.
 */
/**
 * Method newFunctionCategory
 * @param functionCategory
 * @return org.okiproject.serviceapi.roles.FunctionCategory
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.FunctionCategory newFunctionCategory(
    String functionCategory)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * The instance created is local to the application.
 * It may or may not exist in the Roles DB.
 * @param grantPrivilege may be one of the following:
 * @return org.okiproject.serviceapi.roles.GrantPrivilege
 */
/**
 * Method newGrantPrivilege
 * @param grantPrivilege
 * @return org.okiproject.serviceapi.roles.GrantPrivilege
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.GrantPrivilege newGrantPrivilege(
    String grantPrivilege)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * @return the minimum list of Person fields.
 */
/**
 * Method getMinimumPersonFields
 * @return org.okiproject.serviceapi.roles.Field[]
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Field[] getMinimumPersonFields()
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * @return the default list of Person fields.
 */
/**
 * Method getDefaultPersonFields
 * @return org.okiproject.serviceapi.roles.Field[]
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Field[] getDefaultPersonFields()
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * @return the complete list of Person fields.
 */
/**
 * Method getCompletePersonFields
 * @return org.okiproject.serviceapi.roles.Field[]
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.Field[] getCompletePersonFields()
throws Exception {
    throw UNIMPLEMENTED;
}
@return the current list of Person fields.
*/

/**
* Method getCurrentPersonFields
*/
* @return org.okiproject.serviceapi.roles.Field[]
* @throws Exception
*/
public org.okiproject.serviceapi.roles.Field[] getCurrentPersonFields() throws Exception {
throw UNIMPLEMENTED;
}

/**
* Method setCurrentPersonFields
*/
* @param currentFields
* @throws Exception
*/
public void setCurrentPersonFields(
org.okiproject.serviceapi.roles.Field[] currentFields)
throws Exception {
throw UNIMPLEMENTED;
}

/**
* Returns a new instance of Person. The instance is local to the application. The person may or may not exist in the Roles DB. (The key for MIT is KerberosName. Other implementations may use other ways to uniquely identify people.)
*/

/**
* Method newPerson
*/
* @param key
* @return org.okiproject.serviceapi.roles.Person
* @throws Exception
*/
public org.okiproject.serviceapi.roles.Person newPerson(Object key) throws Exception {
throw UNIMPLEMENTED;
}

/**
* Determines if this Person has a matching Person in the Roles DB.
*/
* @param person
* @return true indicates Person is in the Roles DB as are its properties, false indicates Roles DB hasn’t been queried or Person is not in the Roles DB.
*/

/**
* Method isPersonFoundInRoles
*/
* @param person
* @return Boolean
* @throws Exception
*/
public Boolean isPersonFoundInRoles(
org.okiproject.serviceapi.roles.Person person)
throws Exception {
throw UNIMPLEMENTED;
}

/**
* Refresh properties from a changing Roles DB.
*/

/**
* Method refreshPerson
*/
* @param person
* @throws Exception
*/
public void refreshPerson(org.okiproject.serviceapi.roles.Person person)
throws Exception {
throw UNIMPLEMENTED;
}

/**
* Creates a new Qualifier in the Roles DB. This is different from making a new instance of a Qualifier object locally as the Qualifier will be inserted into the Roles DB.
*/

/**
* Method createQualifier
*/
* @param qualifierType
* @param qualifierCode
* @param qualifierDescription
* @param parent
* @return org.okiproject.serviceapi.roles.Qualifier
* @throws Exception
*/
public org.okiproject.serviceapi.roles.Qualifier createQualifier(
org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
throw UNIMPLEMENTED;
}
/**
 * Deletes an existing qualifier in the Roles DB.
 */
/**
 * Method deleteQualifier
 *
 * @param qualifier
 * @throws Exception
 */
public void deleteQualifier(
    org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Determines if this Qualifier has a matching Qualifier in the Roles DB.
 * 
 * @param qualifier
 * @return true indicates Qualifier is in the Roles DB as are its properties.
 *         @return false indicates Roles DB hasn’t been queried or Qualifier is not in the Roles DB.
 */
/**
 * Method isQualifierFoundInRoles
 *
 * @param qualifier
 * @return Boolean
 * @throws Exception
 */
public Boolean isQualifierFoundInRoles(
    org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Refreshes properties from a changing Roles DB.
 */
/**
 * Method refreshQualifier
 *
 * @param qualifier
 * @throws Exception
 */
public void refreshQualifier(
    org.okiproject.serviceapi.roles.Qualifier qualifier)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Creates a new QualifierType in the Roles DB. This is different from making a new instance of a QualifierType object locally as the QualifierType will be inserted into the Roles DB.
 */
/**
 * Method createQualifierType
 *
 * @param type
 * @param description
 * @return org.okiproject.serviceapi.roles.QualifierType
 * @throws Exception
 */
public org.okiproject.serviceapi.roles.QualifierType createQualifierType(
    String type, String description)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Deletes an existing qualifierType in the Roles DB.
 */
/**
 * Method deleteQualifierType
 *
 * @param qualifierType
 * @throws Exception
 */
public void deleteQualifierType(
    org.okiproject.serviceapi.roles.QualifierType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Updates one or more fields in an existing qualifier in the Roles DB.
 */
/**
 * Method updateQualifier
 *
 * @param qualifier
 * @param newQualifierCode
 * @param newDescription
 * @throws Exception
 */
public void updateQualifier(
    org.okiproject.serviceapi.roles.Qualifier qualifier,
    String newQualifierCode, String newDescription)
throws Exception {
    throw UNIMPLEMENTED;
}
public void deleteQualifierType(
    org.okiproject.serviceapi.roles.QualifierType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Determines if this QualifierType has a matching QualifierType in the Roles DB.
 * @param qualifierType
 * @return true indicates QualifierType is in the Roles DB as are its properties.
 * @return false indicates Roles DB hasn't been queried or QualifierType is not in the Roles DB.
 */

***/
* Method isQualifierTypeFoundInRoles
* @param qualifierType
* @return Boolean
* @throws Exception
* */

public Boolean isQualifierTypeFoundInRoles(
    org.okiproject.serviceapi.roles.QualifierType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Refreshes QualifierType properties from a changing Roles DB.
 */

***/
* Method refreshQualifierType
* @param qualifierType
* @throws Exception
* */

public void refreshQualifierType(
    org.okiproject.serviceapi.roles.QualifierType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Updates Description in an existing qualifierType in the Roles DB.
 */

***/
* Method updateQualifierType
* @param qualifier
* @param newDescription
* @throws Exception
* */

public void updateQualifierType(
    org.okiproject.serviceapi.roles.QualifierType qualifierType,
    String newDescription)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Given a Qualifier Type
 * returns the qualifier that is the root of the tree of qualifiers of this type.
 */

***/
* Method getRootQualifier
* @param qualifierType
* @return org.okiproject.serviceapi.roles.Qualifier
* @throws Exception
* */

public org.okiproject.serviceapi.roles.Qualifier getRootQualifier(
    org.okiproject.serviceapi.rolesQUALIFIERType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Given an existing qualifier
 * returns a list of its child qualifiers.
 */

***/
* Method getQualifierChildren
* @param qualifier
* @return org.okiproject.serviceapi.roles.Qualifiers
* @throws Exception
* */

public org.okiproject.serviceapi.roles.Qualifiers getQualifierChildren(
    org.okiproject.serviceapi.roles.ReleaseType qualifierType)
throws Exception {
    throw UNIMPLEMENTED;
}
public org.okiproject.serviceapi.roles.Qualifiers getQualifierChildren(
    org.okiproject.serviceapi.roles.Qualifier qualifier)
    throws Exception {
    throw UNIMPLEMENTED;
}

/*
 * Given a list of selection criteria field/value pairs
 * returns a list of Qualifiers.
 * start indicates first position of first Qualifier in list to return.
 * number indicates the maximum number of Qualifier to return.
 */

/**
 * Method getQualifiersGeneralForm
 *
 * @param selectionCriteria
 * @param start
 * @param number
 *
 * @return org.okiproject.serviceapi.roles.Qualifiers
 *
 * @throws Exception
 */

public org.okiproject.serviceapi.roles.Qualifiers getQualifiersGeneralForm(
    org.okiproject.serviceapi.roles.FieldValue[] selectionCriteria,
    int start, int number)
    throws Exception {
    throw UNIMPLEMENTED;
}
/* instance. */

/**
 * Method newQualifier
 *
 * @param qualifierType
 * @param qualifierCode
 *
 * @return org.okiproject.serviceapi.roles.Qualifier
 *
 * @throws Exception
 *
 */
public org.okiproject.serviceapi.roles.Qualifier newQualifier(
    String qualifierType, String qualifierCode)
throws Exception {
    throw UNIMPLEMENTED;
}

/* Returns a new instance of org.okiproject.serviceapi.roles.Qualifier.
 * The instance may or may not exist in the Roles DB. This is a local
 * instance. */

/**
 * Method newQualifier
 *
 * @param qualifierType
 * @param qualifierCode
 *
 * @return org.okiproject.serviceapi.roles.Qualifier
 *
 * @throws Exception
 *
 */
public org.okiproject.serviceapi.roles.Qualifier newQualifier(
    String qualifierType, String qualifierCode)
throws Exception {
    throw UNIMPLEMENTED;
}

/* Returns a new instance of org.okiproject.serviceapi.roles.QualifierType.
 * The instance may or may not exist in the Roles DB. This is a local
 * instance. */

/**
 * Method newQualifierType
 *
 * @param qualifierType
 *
 * @return org.okiproject.serviceapi.roles.QualifierType
 *
 * @throws Exception
 *
 */
public org.okiproject.serviceapi.roles.QualifierType newQualifierType(
    String qualifierType)
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions
 * of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Model extends java.io.Serializable {
    /** java.sql.Timestamp getEffectiveDate();
    */
    Object getId();
    /** java.sql.Timestamp getExpirationDate();
    */
    String getModifiedBy();
    /** java.sql.Timestamp getModifiedDate();
    */
    java.sql.Timestamp getExpirationDate();
    /**
     * Method getVerbType
     */
    org.okiproject.serviceapi.rules.VerbType getVerbType();
    /**
     * Method getPredicateType
     */
    org.okiproject.serviceapi.rules.PredicateType getPredicateType();
    /**
     * Method getSubjectType
     */
    org.okiproject.serviceapi.rules.SubjectType getSubjectType();
    /**
     * Deletes an existing Model in the Rules DB.
     */
    void delete() throws Exception;
    /**
     * Updates one or more fields in an existing Model in the Rules DB.
     */
    void update(org.okiproject.serviceapi.rules.SubjectType subjectType,
                org.okiproject.serviceapi.rules.VerbType verbType,
                org.okiproject.serviceapi.rules.PredicateType predicateType,
                java.sql.Timestamp effectiveDate, java.sql.Timestamp expirationDate)
                throws Exception;
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Predicate extends java.io.Serializable {

/**
 * Method getId
 */
  Object getId();

/**
 * Method getPredicateType
 */
  org.okiproject.serviceapi.rules.PredicateType getPredicateType();

/**
 * Method getHasChild
 */
  Boolean getHasChild();

/**
 * Method getProperties
 */
  java.util.Properties getProperties();

/**
 * Method setId
 */
  void setId(Object id);

/**
 * Method setPredicateType
 */
  void setPredicateType(
    org.okiproject.serviceapi.rules.PredicateType predicateType);

/**
 * Deletes an existing predicate in the Roles DB.
 */
  void delete() throws Exception;

/**
 * Determines if this Predicate has a matching Predicate in the Rules DB.
 * @return true indicates Predicate is in the Rules DB as are its properties.
 * @return false indicates Rules DB hasn't been queried or Predicate is not in the Rules DB.
 */
  Boolean isFoundInRules() throws Exception;

/**
 * Refreshes properties from a changing Rules DB.
 */
  void refresh() throws Exception;

/**
 * Updates one or more fields in an existing predicate in the Rules DB.
 */
  void update(org.okiproject.serviceapi.rules.PredicateType newPredicateType)
    throws Exception;

/**
 * Adds a parent-child relationship between two existing predicates
 * in the Rules DB.
 */
  void addLink(org.okiproject.serviceapi.rules.Predicate parent)
    throws Exception;

/**
 * Deletes a parent-child relationship between two existing predicates
 * in the Rules DB.
 */
  void deleteLink()
    throws Exception;

/**
 * For an existing predicate, moves the parent link from one existing
 * predicate to another.
 */
*/
void moveLink(org.okiproject.serviceapi.rules.Predicate oldParent,
org.okiproject.serviceapi.rules.Predicate newParent)
throws Exception;

/**
* For an existing predicate, determines if the predicate is the child of
* given parent
*/
//Boolean isChildOf( org.okiproject.serviceapi.rules.Predicate parent ) throws Exception;

/**
* For an existing predicate, get its descendents
*/
//org.okiproject.serviceapi.rules.Predicates getDescendants() throws Exception;
}
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface PredicateType
extends java.io.Serializable {

/**
 * Method getType
 *
 */
String getType();

/**
 * Method setType
 * @param type
 */
void setType(String type);

/**
 * Method isFoundInRules
 * @throws Exception
 */
Boolean isFoundInRules() throws Exception;
}
package org.okiproject.serviceapi.rules;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
<p>The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.</p></e>

* @see org.okiproject.serviceapi.rules.Predicate
* @see org.okiproject.serviceapi.rules.RulesFactory

public interface Predicates
extends java.io.Serializable {
    /**
     * Method hasMorePredicates
     * @throws Exception
     */
    boolean hasMorePredicates() throws Exception;

    /**
     * Method nextPredicate
     * @throws Exception
     */
    org.okiproject.serviceapi.rules.Predicate nextPredicate() throws Exception;
}
package org.okiproject.serviceapi.rules;

/**
   * Copyright (c) 2001 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any
   * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

public interface Rule
extends java.io.Serializable {

/**
   * Method getId
   *
   */
   Object getId();

/**
   * Method getEffectiveDate
   *
   */
   java.sql.Timestamp getEffectiveDate();

/**
   * Method getExpirationDate
   *
   */
   java.sql.Timestamp getExpirationDate();

/**
   * Method getModifiedDate
   *
   */
   java.sql.Timestamp getModifiedDate();

/**
   * Method getModifiedBy
   *
   */
   String getModifiedBy();

/**
   * Method getPredicate
   *
   */
   org.okiproject.serviceapi.rules.Predicate getPredicate();

/**
   * Method getVerb
   *
   */
   org.okiproject.serviceapi.rules.Verb getVerb();

/**
   * Method getPredicateType
   *
   */
   org.okiproject.serviceapi.rules.PredicateType getPredicateType();

/**
   * Method getVerbType
   *
   */
   org.okiproject.serviceapi.rules.VerbType getVerbType();

/**
   * Method getSubjectType
   *
   */
   org.okiproject.serviceapi.rules.SubjectType getSubjectType();

/**
   * Deletes an existing rule in the Rules DB.
   */
   void delete()
   throws Exception;

/**
   * Updates one or more fields in an existing rule
   * in the Rules DB.
   */
   void update(org.okiproject.serviceapi.rules.Subject subject,
               org.okiproject.serviceapi.rules.Verb verb,
               org.okiproject.serviceapi.rules.Predicate predicate,
               org.okiproject.serviceapi.rules.SubjectType subjectType,
               org.okiproject.serviceapi.rules.PredicateType predicateType,
               java.sql.Timestamp effectiveDate,
               java.sql.Timestamp expirationDate)
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

**
* @see org.okiproject.serviceapi.rules.Rule
* @see org.okiproject.serviceapi.rules.RulesFactory

$Source: /cvs/oki/org/okiproject/serviceapi/rules/Rules.java,v $ $Revision: 1.3 $ $Date: 2002/02/08 14:25:47 $
*/

public interface Rules
extends java.io.Serializable {

/**
 * Method hasMoreRules
 * @throws Exception
 */

boolean hasMoreRules()
throws Exception;

/**
 * Method nextRule
 * @throws Exception
 */

guest.org.okiproject.serviceapi.rules.Rule nextRule()
throws Exception;
}
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining
 * a copy of this software and associated documentation files (the "Software"),
 * to deal in the Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense,
 * and/or sell copies of the Software, and to permit persons to whom the Software
 * is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies
 * or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
 * INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE
 * FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

Source: /cvs/oki/org/okiproject/serviceapi/rules/RulesException.java,v $@version@ $Revision: 1.3 $$Date: 2002/02/08 14:25:47 $ */

public class RulesException
    extends java.lang.Exception {
    public static final String QUERYRULEFAILED = "Query rule failed ";
    public static final String CREATERULEFAILED = "Create rule failed ";
    public static final String DELETERULEFAILED = "Delete rule failed ";
    public static final String UPDATERULEFAILED = "Update rule failed ";
    public static final String RULEENUMERATIONFAILED = "Enumeration of rules failed ";
    public static final String NULLARGUMENT = "Null argument ";
    public static final String DEFAULTRULESFAC TORYDIDNOTLOAD = "Default Rules factory did not load ";
    public static final String DATABASESYMBOLICNAMELISTDID NOTLOAD = "Database symbolic name list did not load ";
    public static final String RULESFACTORYPROPERTIESDIDNOTLOAD = "Rules factory properties did not load ";
    public static final String INVALIDSUBJECTKEY = "Invalid subject key ";
    public static final String SUBJECTNOTFOUNDINRULES = "Subject not found in Rules ";
    public static final String SUBJECTTYPENOTFOUNDINRULES = "Subject type not found in Rules ";
    public static final String RULEFIELDDOESNOTEXIST = "Rule field does not exist ";
    public static final String RULEFIELDDOESNOTEXIST = "Rule field does not exist ";
    public static final String VERBIDALREADYEXISTS = "Verb already exists ";
    public static final String INVALIDDATABASE = "Invalid database ";
    public static final String INVALIDVERB = "Invalid verb ";
    public static final String VERBNOTFOUNDINRULES = "Verb not found in Rules ";
    public static final String VERBTYPENOTFOUNDINRULES = "Verb type not found in Rules ";
    public static final String RULEENUMERATIONFAILED = "Enumeration of verbs failed ";
    public static final String INVALIDPREDICATE = "Invalid predicate ";
    public static final String PREDICATENOTFOUNDINRULES = "Predicate not found in Rules ";
    public static final String PREDICATECODENOTFOUNDINRULES = "Predicate code not found in Rules ";
    public static final String PREDICATETYPENOTFOUNDINRULES = "Predicate type not found in Rules ";
    public static final String PREDICATEENUMERATIONFAILED = "Enumeration of predicates failed ";
    public static final String RULESFACTORYPROPERTIESDIDNOTLOAD = "Rules factory properties did not load ";
    public static final String UNIMPLEMENTED = "Unimplemented Rules method ";
    public static final String DEFAULTRULESFAC TORYDIDNOTLOAD = "Default Rules factory did not load ";
    public static final String DATABASESYMBOLICNAMELISTDIDNOTLOAD = "Database symbolic name list did not load ";
    public static final String RULESFACTORYPROPERTIESDIDNOTLOAD = "Rules factory properties did not load ";
    public static final String NULLARGUMENT = "Null argument ";
    public static final String UNIMPLEMENTED = "Unimplemented Rules method ";

    public RulesException(String message) {
        super(message);
    }
}
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

The RulesFactory allows the application developer to create Rules
objects in the Rules database, to get Rules objects from the Rules
database given selection criteria, to ask questions of Rules such as
what subject can do verb for predicate.

The key objects in Rules are Rules, Verbs, Subjects, and
and Predicates. There are some other objects in Rules that play
supporting roles. They are SubjectType, VerbType, PredicateType.

There are two methods to create Rules. One uses method uses
Person, Function, and Qualifier. The other adds effective date and
expiration date. For the method without the dates the effective date
is today and there is no expiration date.

There are methods for getting Rules:

1. getRulesSimpleForm
   This method is overloaded to allow for combinations
   of filtering arguments to find matching Rules.

2. getRulesForDoingVerbWithPredicate
   Given a Verb and a Predicate, finds a list of all
   Rules that would permit a person to do the Verb
   with the Predicate. This method is overloaded
   so that Rules for Verbs within a VerbType
   can be found. This uses a different algorithm than
   getRulesSimpleForm.

   Note that the Predicate in any found Rule
   might be a parent of the Predicate give as an argument.

*/

import java.io.Serializable;

public class RulesFactory implements java.io.Serializable {

  /**
   * Method getDefaultRulesFactory
   * @param agent
   * @return org.okiproject.serviceapi.rules.RulesFactory
   * @throws Exception
   */
  public static org.okiproject.serviceapi.rules.RulesFactory
      getDefaultRulesFactory(
          org.okiproject.serviceapi.common.Agent agent)
      throws Exception {
        java.util.Map properties = getPropertiesFromPRB();
        return
  }

  /**
   * Method getRulesFactory
   * @param agent
   * @param props
   * @return org.okiproject.serviceapi.rules.RulesFactory
   * @throws Exception
   */
  public static org.okiproject.serviceapi.rules.RulesFactory
      getRulesFactory(
          org.okiproject.serviceapi.common.Agent agent,
          java.util.Map props)
      throws Exception {
        java.util.Map properties = props;
if (null == properties) {
  properties = getPropertiesFromPRB();
}

if (null != properties) {
  String factoryClassName = (String) properties.get("RulesFactory");

  if (null != factoryClassName) {
    Class factoryClass = Class.forName(factoryClassName);
    factory.setAgent(agent);
    factory.setProperties(properties);
    return factory;
  }

  throw new org.okiproject.serviceapi.rules.RulesException(
    org.okiproject.serviceapi.rules.RulesException.DEFAULTRULESFAC TORYDIDNOTLOAD);
}

protected static java.util.Properties getPropertiesFromPRB() throws Exception {
  java.util.PropertyResourceBundle prb = (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("RulesFactory");

  if (null != prb) {
    java.util.Properties properties = new java.util.Properties();
    java.util.Enumeration keys = prb.getKeys();
    while (keys.hasMoreElements()) {
      String key = (String) keys.nextElement();
      properties.setProperty(key, (String) prb.handleGetObject(key));
    }
    return properties;
  }

  throw new org.okiproject.serviceapi.rules.RulesException(
    org.okiproject.serviceapi.rules.RulesException.RULESFAC TORYPROPERTIESDIDNOTLOAD);
}

private org.okiproject.serviceapi.common.Agent agent = null;

/**
 * Method getAgent
 *
 * @return org.okiproject.serviceapi.common.Agent
 */
public org.okiproject.serviceapi.common.Agent getAgent() {
  return this.agent;
}

/**
 * Method setAgent
 *
 * @param agent
 */
public void setAgent(org.okiproject.serviceapi.common.Agent agent) {
  this.agent = agent;
}

private java.util.Map properties = null;

/**
 * Method getProperties
 *
 * @return java.util.Map
 */
public java.util.Map getProperties() {
  return this.properties;
}

/**
 * Method setProperties
 *
 * @param properties
 */
public void setProperties(java.util.Map properties) {
  this.properties = properties;
}
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public interface Subject
extends java.io.Serializable {

/**
 * Method getKey
 */
Object getKey();

/**
 * Method setName
 */
String getName();

/**
 * Method getSubjectType
 */
org.okiproject.serviceapi.rules.SubjectType getSubjectType();

/**
 * Method getProperties
 */
java.util.Properties getProperties();

/**
 * Method setKey
 */
void setKey(Object key);

/**
 * Determines if this Subject has a matching Subject in the Rules DB.
 * @return true indicates Subject is in the Rules DB as are its properties.
 * @return false indicates Rules DB hasn’t been queried or Subject is not in the Rules DB.
 */
Boolean isFoundInRules() throws Exception;

/**
 * Refresh properties from a changing Rules DB.
 */
void refresh() throws Exception;
}
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface SubjectType extends java.io.Serializable {

    /**
     * Method getDescription
     *
     * @param description
     */
    void setDescription(String description);

    /**
     * Method getType
     *
     * @param type
     */
    void setType(String type);

    /**
     * Method isFoundInRules
     *
     * @throws Exception
     */
    Boolean isFoundInRules() throws Exception;

    /**
     * Method setDescription
     */

package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

public interface Subjects
    
    extends java.io.Serializable {

    /**
     * Method hasMoreSubjects
     *
     * @throws Exception
     *
     */

    boolean hasMoreSubjects() throws Exception;

    /**
     * Method nextSubject
     *
     * @throws Exception
     *
     */

    org.okiproject.serviceapi.rules.Subject nextSubject() throws Exception;
    
}
package org.okiproject.serviceapi.rules;

/**
 * @see org.okiproject.serviceapi.rules.Verbs
 * @see org.okiproject.serviceapi.rules.VerbType
 * @see org.okiproject.serviceapi.rules.RulesFactory
 */
public interface Verb extends java.io.Serializable {

    /**
     * Method getDescription
     *
     * @return String
    */
    String getDescription();

    /**
     * Method getId
     *
     * @return Object
    */
    Object getId();

    /**
     * Method getModifiedBy
     *
     * @return String
    */
    String getModifiedBy();

    /**
     * Method getModifiedDate
     *
     * @return java.sql.Timestamp
    */
    java.sql.Timestamp getModifiedDate();

    /**
     * Method getName
     *
     * @return String
    */
    String getName();

    /**
     * Method getProperties
     *
     * @return java.util.Properties
    */
    java.util.Properties getProperties();

    /**
     * Method getVerbType
     *
     * @return org.okiproject.serviceapi.rules.VerbType
    */
    org.okiproject.serviceapi.rules.VerbType getVerbType();

    /**
     * Method setId
     *
     * @param id
     */
    void setId(Object id);

    /**
     * Method setName
     *
     * @param name
     */
    void setName(String name);

    /**
     * Method setVerbType
     *
     * @param verbType
     * @throws Exception
     */
    void setVerbType(org.okiproject.serviceapi.rules.VerbType verbType)
        throws Exception;

    /**
     * Deletes an existing Verb in the Rules DB.
     *
     * @throws Exception
     */
    void delete() throws Exception;

    /**
     * Determines if this Verb has a matching Verb in the Rules DB.
     *
     * @return true indicates Verb is in the Rules DB as are its properties.
     * @return false indicates Rules DB hasn't been queried or Verb is not in the Rules DB.
     */
    Boolean isFoundInRules() throws Exception;
}
/**
 * Refreshs properties from a changing Rules DB.
 */
void refresh()
throws Exception;

/**
 * Updates one or more fields in an existing Verb in the Rules DB.
 * @see rules.VerbType
 */
void update(String newName, String newDescription,
             org.okiproject.serviceapi.rules.VerbType newVerbType)
throws Exception;
}
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 */
public interface VerbType extends java.io.Serializable {
    /**
     * Method getDescription
     */
    String getDescription();

    /**
     * Method getType
     */
    String getType();

    /**
     * Method isFoundInRules
     * @throws Exception
     */
    Boolean isFoundInRules() throws Exception;

    /**
     * Method setType
     */
    void setType(String type);
package org.okiproject.serviceapi.rules;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Verbs extends java.io.Serializable {

    /**
     * Method hasMoreVerbs
     * @throws Exception
     */
    boolean hasMoreVerbs() throws Exception;

    /**
     * Method nextVerb
     * @throws Exception
     */
    org.okiproject.serviceapi.rules.Verb nextVerb() throws Exception;
}
package org.okiproject.serviceapi.sched;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

// package org.okiproject.serviceapi.sched;

public interface Location extends java.io.Serializable {}
package org.okiproject.serviceapi.sched;

/**
*p> Copyright (c) 2001 Massachusetts Institute of Technology
<p> Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
<p> The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
<p> THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
<p> $Source: /cvs/oki/org/okiproject/serviceapi/sched/Locations.java,v $
$Revision: 1.2 $ $Date: 2002/02/08 14:25:47 $
*/
public interface Locations
extends java.io.Serializable {

/**
 * Method add
 *
 * @param location
 */
void add(org.okiproject.serviceapi.sched.Location location);

/**
 * Method remove
 *
 * @param location
 */
void remove(org.okiproject.serviceapi.sched.Location location);

/**
 * Method iterator
 */
org.okiproject.serviceapi.sched.LocationsIterator iterator();}
package org.okiproject.serviceapi.sched;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any 
 * person obtaining a copy of this software and associated 
 * documentation files (the "Software"), to deal in the 
 * Software without restriction, including without limitation 
 * the rights to use, copy, modify, merge, publish, 
 * distribute, sublicense, and/or sell copies of the 
 * Software, and to permit persons to whom the Software is 
 * furnished to do so, subject to the following conditions:
 * 
The above copyright notice and this permission notice 
shall be included in all copies or substantial portions of 
the Software.
 * 
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY 
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE 
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR 
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE 
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, 
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF 
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN 
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS 
IN THE SOFTWARE.
 */

public interface LocationsIterator
extends java.io.Serializable {

/**
 * Method hasNext
 * @return boolean
 */
    Boolean hasNext();

/**
 * Method next
 * @return org.okiproject.serviceapi.sched.Location
 */
    org.okiproject.serviceapi.sched.Location next();
}
package org.okiproject.serviceapi.sched;

/**
 * Purpose: Represents the schedule abstraction
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
* Software without restriction, including without limitation
* the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
* Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
* The above copyright notice and this permission notice
* shall be included in all copies or substantial portions of
* the Software.
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
* KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
* WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
* PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
* CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
* CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
* IN THE SOFTWARE.
* $Source: /cvs/oki/org/okiproject/serviceapi/sched/Schedule.java,v $
* $Revision: 1.10 $
* $Date: 2002/02/11 00:54:52 $
*/

public interface Schedule extends java.io.Serializable {

  /**
   * Title of the schedule. A Schedule is a named entity.
   */
  public String getSchedName();

  /**
   * Method setSchedName
   */
  public void setSchedName(final String schedName);

  /**
   * Method setAgent -- sets the owner of this Schedule
   */
  public void setAgent(final org.okiproject.serviceapi.common.Agent agent);

  /**
   * @return agent -- gets the agent associated with this Schedule
   */
  public org.okiproject.serviceapi.common.Agent getAgent();

  public org.okiproject.serviceapi.sched.ScheduleItemQueue getScheduleItems();

  /**
   * Returns list of items successfully/unsuccessfully scheduled?? */
  public org.okiproject.serviceapi.sched.ScheduleItemQueue requestSchedule( org.
  okiproject.serviceapi.common.Agent[] others, Location location, TimeSpan timespan );

  /**
   * Returns list of items successfully/unsuccessfully canceled?? */
  /**
   * Does there need to be a status and a priority associated with each
   * ScheduleItem? Yes. I'll put that in.
   * This way, we can examine the state of each to determine the conflictin
   * g items.
   */
  public org.okiproject.serviceapi.sched.ScheduleItemQueue cancelSchedule( org.o
  kiproject.serviceapi.common.Agent[] others, Location location, TimeSpan timespan);

  //public boolean setTaskList(  org.okiproject.serviceapi.sched.TaskQueue taskQ );
}
package org.okiproject.serviceapi.sched;

/**
 * Purpose: Represents the schedule abstraction
 * <p/>
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * <p/>
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * <p/>
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * <p/>
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 * <p/>
 * $Source: /cvs/ok/ok.org/okiproject/serviceapi/sched/ScheduleItem.java,v $
 * $Revision: 1.4 $
 * $Date: 2002/02/11 00:54:52 $
 */

public interface ScheduleItem
    extends java.io.Serializable {
    public static final String CANCELED = "CANCELED";
    public static final String ACTIVE = "ACTIVE";
    public static final String INACTIVE = "INACTIVE";

    /**
     * Title of the schedule item. A Scheduled Item is a named entity.
     *
     **/
    public String getItemName();

    /**
     * Method setItemName
     * @param itemName
     *
     **/
    public void setItemName(final String itemName);

    /**
     * Method getLocation
     *
     **/
    public org.okiproject.serviceapi.sched.Location getLocation();

    /**
     * Method setLocation
     * @param location
     *
     **/
    public boolean setLocation(
        org.okiproject.serviceapi.sched.Location location);

    /**
     * Method getTimeSpan
     *
     **/
    public org.okiproject.serviceapi.sched.TimeSpan getTimeSpan();

    /**
     * Method setTimeSpan
     * @param when
     *
     **/
    public boolean setTimeSpan(
        org.okiproject.serviceapi.sched.TimeSpan when);

    /**
     * Method unsetTimeSpan
     * @param span
     *
     **/
    public void unsetTimeSpan(org.okiproject.serviceapi.sched.TimeSpan span);

    /**
     * Method getPriority
     *
     **/
    public int getPriority();

    /**
     * Method setPriority
     * @param priority
     *
     **/
    public void setPriority(int priority);
public void setStatus(final String status);

/**
 * Method getStatus
 * 
 */

public String getStatus();

//public org.okiproject.serviceapi.ScheduleItemIterator getScheduleItemIterator();
package org.okiproject.serviceapi.sched;

/**
 * Purpose: Represents a queue of ScheduleItem's.
 */
public interface ScheduleItemQueue extends java.io.Serializable {

    /**
     * Add the specified task to the queue.
     */
    public void add(final org.okiproject.serviceapi.sched.ScheduleItem item);

    /**
     * Remove the specified task from the queue.
     */
    public void remove(final org.okiproject.serviceapi.sched.ScheduleItem item);

    /**
     * Get the default iterator. Possibly a simple FIFO traversal.
     */
    public ScheduleItemQueueIterator getIterator();
}
package org.okiproject.serviceapi.sched;

/**
 * Purpose: Abstracts the navigation of ScheduleItems.
 * <p>Copyright (c) 2001 Massachusetts Institute of Technology
 * <p>Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * <p>The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * <p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 * <p>$Source: /cvs/okiproject/serviceapi/sched/ScheduleItemQueueIterator.java,v $
 * $Log: ScheduleItemQueueIterator.java,v $
 * $Revision: 1.2 $
 * $Date: 2002/02/08 14:25:47 $
 */
 public interface ScheduleItemQueueIterator
 extends java.io.Serializable {

 /**
 * Return the next ScheduleItem in the queue.
 **/
 public org.okiproject.serviceapi.sched.ScheduleItem next();

 /**
 * Is there another?
 **/
 public Boolean hasNext();
}
package org.okiproject.serviceapi.sched;

/**
 * Purpose: Represents an interval in time.
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface TimeSpan
extends java.io.Serializable {

/**
 * Get the start time.
 * @return
 */
public long getStartTime();

/**
 * Get the end time.
 */
public long getEndTime();

/**
 * Convenience method to get the interval.
 */
public long getInterval();

/**
 * Set a start time.
 * @param start Represents the start date/time
 */

// Caveat: We will not violate encapsulation here. Let the ctor
// provide the end points. Disallow reset?
//public void setStartTime( long time );

//public void setEndTime( long time );
package org.okiproject.serviceapi.sql;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Blob extends java.io.Serializable {

    /**
     * Method getBytes
     * @param pos
     * @param len
     * @throws Exception
     */
    byte[] getBytes(Long pos, Integer len)
    throws Exception;

    /**
     * Method length
     */
    long length();

    /**
     * Method position
     * @param pattern
     * @param start
     * @throws Exception
     */
    Long position(byte[] pattern, Long start)
    throws Exception;
}
package org.okiproject.serviceapi.sql;

/**
   * Copyright (c) 2001 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any
   * person obtaining a copy of this software and associated
   * documentation files (the "Software"), to deal in the
   * Software without restriction, including without limitation
   * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
   * Software, and to permit persons to whom the Software is
   * furnished to do so, subject to the following conditions:
   * 
   * The above copyright notice and this permission notice
   * shall be included in all copies or substantial portions of
   * the Software.
   * 
   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
   * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
   * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
   * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
   * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
   * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
   * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
   * IN THE SOFTWARE.
   * 
   * @Source: /cvs/oki/org/okiproject/serviceapi/sql/Clob.java,v
   * @version 1.3
   * @date 2002/02/08 14:25:47
   *
   *
   */

public interface Clob
extends java.io.Serializable {

    /**
     * Method getSubString
     * @param pos
     * @param length
     * @throws Exception
     */
    String getSubString(Long pos, Integer length)
    throws Exception;

    /**
     * Method length
     */
    long length();

    /**
     * Method position
     * @param searchstr
     * @param start
     * @throws Exception
     */
    Long position(String searchstr, Long start)
    throws Exception;
}
package org.okiproject.serviceapi.sql;

/**<p> Copyright (c) 2001 Massachusetts Institute of Technology</p>Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.</p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.</p>**Source: /cvs/oki/org/okiproject/serviceapi/sql/ColumnMetaData.java,v $@version@ $Revision: 1.3 $$Date: 2002/02/08 14:25:47 $*/
public interface ColumnMetaData
extends java.io.Serializable {

/**
 * Method getCatalogName
 */
String getCatalogName();

/**
 * Method getColumn
 */
int getColumn();

/**
 * Method getColumnClassName
 */
String getColumnClassName();

/**
 * Method getColumnDisplaySize
 */
int getColumnDisplaySize();

/**
 * Method getColumnName
 */
String getColumnName();

/**
 * Method getColumnType
 */
int getColumnType();

/**
 * Method getColumnTypeName
 */
String getColumnTypeName();

/**
 * Method getPrecision
 */
int getPrecision();

/**
 * Method getScale
 */
int getScale();

/**
 * Method getSchemaName
 */
String getSchemaName();

/**
 * Method getTableName
 */
String getTableName();

/**
 * Method isAutoIncrement
 */
boolean isAutoIncrement();

/**
 * Method isCaseSensitive
 */
boolean isCaseSensitive();

/**
 * Method isCurrency
 */
boolean isCurrency();

/**
 * Method isDefinitelyWritable
 */

boolean isDefinitelyWritable();

/**
 * Method getNullable
 */
int getNullable();

/**
 * Method isReadOnly
 */
boolean isReadOnly();

/**
 * Method isSearchable
 */
boolean isSearchable();

/**
 * Method isSigned
 */
boolean isSigned();

/**
 * Method isWritable
 */
boolean isWritable();
}
package org.okiproject.serviceapi.sql;

public interface Connection extends java.io.Serializable {

/**
 * Method getDatabase
 */
String getDatabase();

/**
 * Method getUsername
 */
String getUsername();

/**
 * Method getId
 */
String getId();

/**
 * Method getAutoCommit
 */
Boolean getAutoCommit();

/**
 * Method setAutoCommit
 *
 * @param autoCommit
 */
void setAutoCommit(Boolean autoCommit);

/**
 * Method getProcedures
 */

org.okiproject.serviceapi.sql.Table getProcedures(String catalog,
        String schemaPattern, String procedureNamePattern)
        throws Exception;

/**
 * Method getProcedureColumns
 */

org.okiproject.serviceapi.sql.Table getProcedureColumns(String catalog,
        String schemaPattern, String procedureName,
        String columnNamePattern)
        throws Exception;

/**
 * Method executeQuery
 */

org.okiproject.serviceapi.sql.Table executeQuery(String sql)
        throws Exception;

/**
 * Method executeQuery
 */

org.okiproject.serviceapi.sql.Table executeQuery(String sql, Object[] objs)
        throws Exception;

/**
 * Method executeQuery
 */

org.okiproject.serviceapi.sql.Table executeQuery(String sql, Object[] objs)
        throws Exception;
int[] datatypes)
throws Exception;
/**
 * Method connect
 * @throws Exception
 */
void connect()
throws Exception;

/**
 * Method disconnect
 * @throws Exception
 */
void disconnect()
throws Exception;

/**
 * Method commit
 * @throws Exception
 */
void commit()
throws Exception;

/**
 * Method rollback
 * @throws Exception
 */
void rollback()
throws Exception;
}
package org.okiproject.serviceapi.sql;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 **
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 **
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 **
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 **
 ** Source: /cvs/uki/org/okiproject/serviceapi/sql/ConnectionException.java,v $ 
 @version 1.3 $ 
 $date: 2002/02/08 14:25:47 $ 
 ***/

public class ConnectionException extends Exception {
  public static final String INVALIDDRIVERNAME = "Invalid jdbc driver name\n";
  public static final String INVALIDDATABASEURL = "Invalid database URL\n";
  public static final String INVALIDDATABASE = "Invalid database name\n";
  public static final String INVALIDUSERNAME = "Invalid user name\n";
  public static final String INVALIDPASSWORD = "Invalid password\n";
  public static final String INVALIDARGUMENTS = "Invalid arguments\n";

  /**
   * Constructor ConnectionException
   *
   * @param message
   *
   */
  public ConnectionException(String message) {
    super(message);
  }
}
package org.okiproject.serviceapi.sql;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

package org.okiproject.serviceapi.sql;

public interface Rows extends java.io.Serializable {

    /**
     * Method getColumnNames
     * @throws Exception
     */
    String[] getColumnNames() throws Exception;

    /**
     * Method getColumn
     * @param columnName
     * @throws Exception
     */
    int getColumn(String columnName) throws Exception;

    /**
     * Method getRowsMetaData
     */
    org.okiproject.serviceapi.sql.RowsMetaData getRowsMetaData();

    /**
     * Method hasMoreRows
     * @throws Exception
     */
    boolean hasMoreRows() throws Exception;

    /**
     * Method nextRow
     * @throws Exception
     */
    Object[] nextRow() throws Exception;
}
package org.okiproject.serviceapi.sql;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 *
 * $Source: /cvs/oki/org/okiproject/serviceapi/sql/RowsMetaData.java,v $
 * $Revision: 1.3 $
 * $Date: 2002/02/08 14:25:47 $
 */

public interface RowsMetaData
extends java.io.Serializable {

/**
 * Method getCount
 *
 */
int getColumnCount();

/**
 * Method getColumnsMetaData
 *
 */
org.okiproject.serviceapi.sql.ColumnMetaData[] getColumnsMetaData();
}
package org.okiproject.serviceapi.sql;

/**
* Copyright (c) 2001 Massachusetts Institute of Technology
* Permission is hereby granted, free of charge, to any
* person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
* The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
* KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
* WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
* PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
* CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
* CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
* IN THE SOFTWARE.
*/

Source: /cvs/oki/org/okiproject/serviceapi/sql/SqlFactory.java,v $@version@ $Revision: 1.4 $$Date: 2002/02/08 23:09:10 $/*
public class SqlFactory
implements java.io.Serializable {
    private static final org.okiproject.serviceapi.sql.SqlFactoryException UNIMPLEMENTED =
        new org.okiproject.serviceapi.sql.SqlFactoryException(  
            org.okiproject.serviceapi.sql.SqlFactoryException.UNIMPLEMENTED);

    /**
     * Method getDefaultSqlFactory
     * @param agent
     * @throws Exception
     */
    public static org.okiproject.serviceapi.sql.SqlFactory getDefaultSqlFactory(
        org.okiproject.serviceapi.common.Agent agent, java.util.Map props)
        throws Exception {
            java.util.Map properties = props;
            if (null == properties) {
                properties = getPropertiesFromPRB();
            }
            if (null != properties) {
                String factoryClassName = (String) properties.get("SqlFactory");
                if (null != factoryClassName) {
                    org.okiproject.serviceapi.sql.SqlFactory factory =  
                        (org.okiproject.serviceapi.sql.SqlFactory) Class.forName(            factoryClassName).newInstance();
                    factory.setProperties(properties);
                    return factory;
                }
            }
            throw new org.okiproject.serviceapi.sql.SqlFactoryException(
                org.okiproject.serviceapi.sql.SqlFactoryException.DEFAULTSQLFACTORYDIDNOTLOAD);
        }

    private org.okiproject.serviceapi.common.Agent agent = null;
    protected org.okiproject.serviceapi.common.Agent getAgent() {
        return this.agent;
    }

    protected void setAgent(org.okiproject.serviceapi.common.Agent agent) {
        this.agent = agent;
    }

    protected void setProperties(java.util.Map properties) {
        this.properties = properties;
    }

    protected static java.util.Properties getPropertiesFromPRB()  
        throws Exception {
        java.util.PropertyResourceBundle prb =  
            (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle(  
                "SqlFactory");
        if (null != prb) {
            java.util.Properties properties = new java.util.Properties();
            java.utilEnumeration keys = prb.getKeys();
            while (keys.hasMoreElements()) {
                String key = (String) keys.nextElement();
                properties.setProperty(key, (String) prb.handleGetObject(key));
            }
        }

    }
}
return properties;
}

throw new org.okiproject.serviceapi.sql.SqlFactoryException(
    org.okiproject.serviceapi.sql.SqlFactoryException.SQLFACTORYPROPERTIESDIDNOTLOAD
);

/**
 * Method newConnection
 * @param databaseURL
 * @param driverName
 * @param username
 * @param password
 * @return org.okiproject.serviceapi.sql.Connection
 * @throws Exception
 */
public org.okiproject.serviceapi.sql.Connection newConnection(
    String databaseURL, String driverName, String username, String password)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Method newConnection
 * @param database
 * @param username
 * @param password
 * @return org.okiproject.serviceapi.sql.Connection
 * @throws Exception
 */
public org.okiproject.serviceapi.sql.Connection newConnection(
    String database, String username, String password)
throws Exception {
    throw UNIMPLEMENTED;
}

/**
 * Method newConnection
 * @param props
 * @return org.okiproject.serviceapi.sql.Connection
 * @throws Exception
 */
public org.okiproject.serviceapi.sql.Connection newConnection(
    java.util.Map props)
throws Exception {
    throw UNIMPLEMENTED;
}
package org.okiproject.serviceapi.sql;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/okiproject/serviceapi/sql/SqlFactoryException.java,v $
$Revision: 1.3 $
$Date: 2002/02/08 14:25:47 $
*/

public class SqlFactoryException
extends Exception {

public static final String UNIMPLEMENTED                  =
"Unimplemented Sql method ";

public static final String DEFAULTSQLFACTORYDIDNOTLOAD    =
"Default Sql factory did not load ";

public static final String SQLFACTORYPROPERTIESDIDNOTLOAD =
"Sql factory properties did not load ";

/**
 * Constructor SqlFactoryException
 *
 * @param message
 *
 */

public SqlFactoryException(String message) {
    super(message);
}
}
package org.okiproject.serviceapi.sql;

/**
  * Copyright (c) 2001 Massachusetts Institute of Technology
  *
  * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
  *
  * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
  *
  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
  */

public interface Table
extends java.io.Serializable {

  /**
   * Method getRows
   */

  org.okiproject.serviceapi.sql.Rows getRows();}
package org.okiproject.serviceapi.sql;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

* The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

* Source: /cvs/oki/org/okiproject/serviceapi/sql/TableException.java,v $ @version@ $Revision: 1.3 $ $Date: 2002/02/08 14:25:47 $ */

public class TableException
extends Exception {
    public static final String GETCOLUMNFAILED = "Get column data failed\n";
    public static final String GETROWFAILED = "Get row data failed\n";
    public static final String GETROWSFAILED = "Get rows failed\n";
    public static final String GETROWMETADATAFAILED = "Get row metadata failed\n";
    public static final String GETTABLEFAILED = "Get table failed\n";
    public static final String BLOBGETBYTESFAILED = "Blob get bytes failed\n";
    public static final String BLOBPOSITIONFAILED = "Blob position failed\n";
    public static final String CLOBGETSUBSTRINGFAILED = "Clob get substring failed\n";
    public static final String CLOBPOSITIONFAILED = "Clob position failed\n";

    /**
     * Constructor TableException
     * @param message
     */
    public TableException(String message) {
        super (message);
    }
}
package org.okiproject.serviceapi.test.authen;

/**
 * @author
 * @version
 */
public class authen_api_example {

/**
 * Constructor authen_api_example
 *
 */
public authen_api_example(java.io.PrintStream out) {
        try {

            /** Get the MIT implementation of the Authen API */
            org.okiproject.serviceapi.authen.AuthenFactory myFactory =
                    org.okiproject.serviceapi.authen.AuthenFactory.getDefaultAuthenFactory(null);

            String[] types = myFactory.getAuthenTypes();

            if (null != types) {
                for (int i = 0; types.length != i; i++) {
                        out.println(types[i]);
                        System.err.println(types[i]);
                }
            }

            out.println("end of Authen types");

            /** Get an authenticated user */
            org.okiproject.serviceapi.common.Agent user = myFactory.getAgent();

            if (null != user) {
                java.util.Properties userProperties =
                        ((org.okiproject.serviceapi.common.Person) user).getProperties();

                java.util.Iterator propertiesKeys = userProperties.keySet().iterator();

                while (propertiesKeys.hasNext()) {
                    Object key = propertiesKeys.next();
                    out.println("Key:" + key + " value:" + userProperties.get(key));
                }

            } out.println("end of Authenticated user");

            if (null != user) {
                String kerberos = null;

                /** Get an authentication types for authenticated user */
                types = myFactory.getAuthenTypes(user);

                if (null != types) {
                    for (int i = 0; types.length != i; i++) {
                        if (-1 != types[i].toLowerCase().indexOf("kerberos")) {
                            kerberos = types[i];
                        }
                        out.println(types[i]);
                    }
                }

            } out.println("end of Authen types");
        }
    } catch (Exception e) {
    }
}

public static void main(String[] args) {
    try {

        /** Get the MIT implementation of the Authen API */
        org.okiproject.serviceapi.authen.AuthenFactory myFactory =
                org.okiproject.serviceapi.authen.AuthenFactory.getDefaultAuthenFactory(null);

        String[] types = myFactory.getAuthenTypes();

        if (null != types) {
            for (int i = 0; types.length != i; i++) {
                    out.println(types[i]);
            }
        }
    }
}
org/okiproject/serviceapi/test/authen/authen_api_example.java

out.println("end of Authen types for Authenticated user");

/******************************************************************************
** Destroy all authentication for authenticated user
******************************************************************************/
out.println("Destroy all authentication for Authenticated user:");
myFactory.destroyAuthen(user);
types = myFactory.getAuthenTypes(user);
if (null != types) {
    for (int i = 0; types.length != i; i++) {
        out.println(types[i]);
    }
}
out.println("end of Destroy all authentication for Authenticated user");

/******************************************************************************
** Get an authenticated user with a specific authentication type
******************************************************************************/
out.println("Get an Authenticated user with authentication type of KERBEROS4:");
try {
    user = myFactory.getAgent(user, kerberos);
    out.println("user.getName()": user.getName());
    out.println(user);
    if (null != user) {
        java.util.Properties userProperties =
            (org.okiproject.serviceapi.common.Person) user).getProperties();
        java.util.Iterator propertiesKeys =
            userProperties.keySet().iterator();
        while (propertiesKeys.hasNext()) {
            Object key = propertiesKeys.next();
            out.println("Key:" + key + " value:" + userProperties.get(key));
        }
    }
    types = myFactory.getAuthenTypes(user);
    if (null != types) {
        for (int i = 0; types.length != i; i++) {
            out.println(types[i]);
        }
    }
    catch (Exception e) {
        out.println("Exception: " + e);
    }
}
out.println("end of Get an Authenticated user with authentication type of KERBEROS4");
if (null != user) {
    try {
        user = myFactory.getAgent(user, kerberos);
        out.println("user.getName()": user.getName());
        out.println(user);
        if (null != user) {
            java.util.Properties userProperties =
                (org.okiproject.serviceapi.common.Person) user).getProperties();
            java.util.Iterator propertiesKeys =
                userProperties.keySet().iterator();
            while (propertiesKeys.hasNext()) {
                Object key = propertiesKeys.next();
                out.println("Key:" + key + " value:" + userProperties.get(key));
            }
        }
        types = myFactory.getAuthenTypes(user);
        if (null != types) {
            for (int i = 0; types.length != i; i++) {
                out.println(types[i]);
            }
        }
        catch (Exception e) {
            out.println("Exception: " + e);
        }
    }
    out.println("end of Get another Authenticated user with authentication type of KERBEROS4");
}
if (null != user) {
    try {
        myFactory.destroyAuthen(user, kerberos);
        types = myFactory.getAuthenTypes(user);
        if (null != types) {
            for (int i = 0; types.length != i; i++) {
                out.println(types[i]);
            }
        }
    }
    catch (Exception e) {
        out.println("Exception: " + e);
    }
}
out.println("Destroy authentication of type KERBEROS4 for Authenticated user:");
try {
    myFactory.destroyAuthen(user, kerberos);
    types = myFactory.getAuthenTypes(user);
    if (null != types) {
        for (int i = 0; types.length != i; i++) {
            out.println(types[i]);
        }
    }
    catch (Exception e) {
        out.println("Exception: " + e);
    }
}
out.println("end of Destroy authentication of type KERBEROS4 for Authenticated user");

/******************************************************************************
** Get another authenticated user with a specific authentication type
******************************************************************************/
out.println("Get another Authenticated user with authentication type of KERBEROS4:");

/**
 * Destroy authentication array of types for authenticated user
 */

out.println("Destroy authentication array of types for Authenticated user:");

try {
    types = myFactory.getAuthenTypes(user);
    myFactory.destroyAuthen(user, types);
    types = null;
} catch (Exception e) {
    out.println("Exception: " + e);
    out.println("end of Destroy authentication array of types for Authenticated user");
}

} 

catch (Throwable exx) {
    out.println(exx.getMessage());
    exx.printStackTrace();
}

/**
 * Method main
 *
 * @param args
 */

public static void main(String[] args) {
    new org.okiproject.serviceapi.test.authen.authen_api_example((java.io.PrintStream) System.out);
    System.exit(0);
}
/**
   * Copyright (c) 2002 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
   *
   * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
   *
   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
   */

public class rmi_client_example {

    /**
     * Method main
     * @param args
     */
    public static void main(String[] args) {
        try {
            /*********************************************************
            ** Get the MIT implementation of the Authen API
            ***********************************************************/
            org.okiproject.serviceapi.authen.AuthenFactory myFactory =
                    org.okiproject.serviceapi.authen.AuthenFactory.getDefaultAuthenFactory();

            /*********************************************************
            ** Get an authenticated user
            ***********************************************************/
            System.out.println("Authenticated user:");
            org.okiproject.serviceapi.common.Agent user = myFactory.getAgent();

            System.out.println(user.getName());
            System.out.println("end of Authenticated user");
            if (null != user) {
                /*********************************************************
                ** Get the MIT implementation of the Rmi Client API
                ***********************************************************/
                org.okiproject.serviceapi.rmi.ClientFactory clientFactory =
                        org.okiproject.serviceapi.rmi.ClientFactory.getDefaultClientFactory();

                /*********************************************************
                ** Get an RMI Client
                ***********************************************************/
                org.okiproject.serviceapi.rmi.Client client =
                        clientFactory.createClient("blue-geese", new Integer(1099), "rcmd",
                                "org.okiproject.serviceimpl.mit.rmisl.Server", user);

                /*********************************************************
                ** Test the RMI Client
                ***********************************************************/
                System.out.println("Test the RMI Client");
                System.out.println(client.test());
                Class[] argTypes = new Class[0];
                Object[] arguments = new Object[0];
                String testString = "upper case this string";

                System.out.println(testString + ":");
                + client.invoke(testString, "toUpperCase", argTypes, arguments));
            } catch (Throwable exx) {
                System.out.println(exx.getMessage());
                exx.printStackTrace();
            }
            System.exit(0);
        } catch (Throwable exx) {
            System.out.println(exx.getMessage());
            exx.printStackTrace();
        }
    }
}
/**
   * Copyright (c) 2002 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any
   * person obtaining a copy of this software and associated
   * documentation files (the "Software"), to deal in the
   * Software without restriction, including without limitation
   * the rights to use, copy, modify, merge, publish, distribute,
   * sublicense, and/or sell copies of the
   * Software, and to permit persons to whom the Software is
   * furnished to do so, subject to the following conditions:
   * The above copyright notice and this permission notice
   * shall be included in all copies or substantial portions of
   * the Software.
   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
   * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
   * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
   * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
   * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
   * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
   * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
   * IN THE SOFTWARE.
   */

public class rmi_server_example {

    /**
     * Method main
     * @param args
     */

    public static void main(String[] args) {
        try {
            // Sample pseudo-application demonstrating Rmi Server API
            // 01/25/2002
            //**************************************************************************
            // Get the MIT implementation of the Rmi Server API
            //**************************************************************************
            org.okiproject.serviceapi.rmi.ServerFactory serverFactory =
                    org.okiproject.serviceapi.rmi.ServerFactory.getDefaultServerFactory();
        }
        catch (Throwable exx) {
            System.out.println(exx.getMessage());
            exx.printStackTrace();
        }
    }
}
public class roles_api_example {

    // Set some constants
    static final Boolean TRUE = new Boolean(true);
    static final Boolean FALSE = new Boolean(false);

    /**
     * Method main
     * @param args
     */
    public static void main(String[] args) {
        try {
            // Show the user the list of symbolic DB Names.
            String[] symbolicDBnameChoices = org.okiproject.serviceapi.roles.RolesFactory.getDatabaseSymbolicNames();
            System.out.println("Symbolic DB name choices:");
            for (int i = 0; symbolicDBnameChoices.length != i; i++) {
                System.out.println("     "+symbolicDBnameChoices[i]);
            }

            System.out.println("end of name choices");

            // Suppose the user chooses symbolic DB Name "prod_roles".
            // (Choices might be production, test, etc.)
            // Get the MIT implementation of the Roles interface (currently, your only
            // choice) and pass it a service name (e.g. "Roles") that this application
            // uses to identify this instance of roles and the symbolic DB Name.

            org.okiproject.serviceapi.roles.Person person1 = myFactory.newPerson("repa");
            org.okiproject.serviceapi.roles.FunctionCategory category1 = myFactory.newFunctionCategory("GRAD");
            org.okiproject.serviceapi.roles.Authorizations authList1 = myFactory.getAuthsSimpleForm(person1,
                category1,
                TRUE,
                FALSE,
                TRUE)

            System.out.println(person1.getKey() + " has the following authorizations in category: "+category1.getCategory());

            System.out.println(authList1.hasMoreAuthorizations()) {
                // Print the list of auths.
                System.out.println(authList1.nextAuthorization());
            }

            System.out.println("end of authorizations for "+person1.getKey());

            while (authList1.hasMoreAuthorizations()) {
                // Print the list of auths.
                System.out.println(authList1.nextAuthorization());
            }
        }
    }
}
try {
    Boolean heCanDoIt = myFactory.isAuthorized(person2, function2, qualifier2);
    if (heCanDoIt.booleanValue()) {
        System.out.println(person2.getKey() + " can do Function " + function2.getName() + " with Qualifier " + qualifier2.getCode() + " with type " + qualifier2.getQualifierType().getType());
        if (null != people) {
            while (people.hasMorePeople()) {
                org.okiproject.serviceapi.roles.People.person = people.nextPerson();
                System.out.println(" person: " + person.getKey());
            }
        }
        System.out.println("end of list of people");
        System.out.println();
    } else {
        System.out.println(person2.getKey() + " is out of luck if he wants to do Function " + function2.getName() + " without qualifier " + qualifier2.getCode() + " with type " + qualifier2.getQualifierType().getType());
    }
    System.out.println();
} catch (Exception ex) {
    System.out.println(ex.getMessage());
}
/****************************************************************************/
org.okiproject.serviceapi.roles.Person person4a = myFactory.newPerson("cshubert");
function4a = myFactory.newFunction("TEST", "TESTFUN1");
qualifier4a = myFactory.newQualifier("ORGU", "001702");
authList4a = myFactory.getAuthsSimpleForm(person4a, // 1st argument is a person
    function4a, // 2nd argument is a function
    qualifier4a, // 3rd argument is a qualifier
    TRUE, // Let’s only find auths.
    that are active today
    FALSE, // Let's not expand Functions
    TRUE // Let's expand Qualifier
);    try {
    System.out.println("Delete active authorizations");
    while (authList4a.hasMoreAuthorizations()) {
        org.okiproject.serviceapi.roles.Authorization auth4a = authList4a.nextAuthorization();
        System.out.println("Delete active authorization: " + auth4a);
        auth4a.delete();
    }
} catch (Exception ex4a) {
    System.out.println("delete active authorization failure: " + ex4a.getMessage());
}
    // Print the list of people
    org.okiproject.serviceapi.roles.People.person = people.nextPerson();
    System.out.println(" person: " + person.getKey());
}
/****************************************************************************/
org.okiproject.serviceapi.roles.Role person4a = null;
org.okiproject.serviceapi.roles.Function function4a = null;
org.okiproject.serviceapi.roles.Qualifier qualifier4a = null;
org.okiproject.serviceapi.roles.Authorizations authList4a = null;
/****************************************************************************/
org.okiproject.serviceapi.roles.Role function3 = myFactory.newFunction("SAP", "CAN SPEND OR COMMIT FUNDS");
org.okiproject.serviceapi.roles.Qualifier qualifier3 = myFactory.newQualifier("F1652700");
org.okiproject.serviceapi.roles.People people = myFactory.getWhoCanDo(function3, qualifier3, TRUE);
System.out.println("People who can do function " + function3.getName() + " in category " + function3.getFunctionCategory().getCategory() + " without qualifier " + qualifier3.getCode() + " with type " + qualifier3.getQualifierType().getType());
    if (null != people) {
        while (people.hasMorePeople()) {
            org.okiproject.serviceapi.roles.People.person = people.nextPerson();
            System.out.println(" person: " + person.getKey());
        }
    }
    System.out.println("end of list of people");
    System.out.println();
/****************************************************************************/
org.okiproject.serviceapi.roles.Role person4a = null;
org.okiproject.serviceapi.roles.Function function4a = null;
org.okiproject.serviceapi.roles.Qualifier qualifier4a = null;
org.okiproject.serviceapi.roles.Authorizations authList4a = null;
/****************************************************************************/
org.okiproject.serviceapi.roles.Role person4a = null;
org.okiproject.serviceapi.roles.Function function4a = null;
org.okiproject.serviceapi.roles.Qualifier qualifier4a = null;
org.okiproject.serviceapi.roles.Authorizations authList4a = null;
/****************************************************************************/
System.out.println("end of delete inactive authorizations");
System.out.println();
}
catch (Exception ex4aa) {
    System.out.println("Delete inactive authorization failure: "+ ex4aa.getMessage());
}

/******************************************************************************
** Create a new Authorization
*******************************************************************************/
org.okiproject.serviceapi.roles.Person person4 = myFactory.newPerson("cshubert");
org.okiproject.serviceapi.roles.Function function4 = myFactory.newFunction("TEST", "TESTFUN1");
org.okiproject.serviceapi.roles.GrantPrivilege priv4 = myFactory.newGrantPrivilege("ORGU", "001702");
org.okiproject.serviceapi.roles.GrantPrivilege qualifier4 =
    org.okiproject.serviceapi.roles.GrantPrivilege.NOTGRANTABLE;
org.okiproject.serviceapi.roles.Authorization authorization4 = null;
try {
    authorization4 = myFactory.createAuthorization(person4, function4, qualifier4, priv4, TRUE);
    System.out.println(person4.getKey() + " is now authorized to do Function " + function4.getName() + " with qualifier " + qualifier4.getCode() + " with type " + qualifier4.getQualifierType().getType() + "\nstarting on " + authorization4.getEffectiveDate());
    System.out.println();
} catch (Exception ex4) {
    System.out.println("create authorization failure: " + ex4.getMessage());
}

/******************************************************************************
** Update an Authorization
*******************************************************************************/
priv4 = myFactory.newGrantPrivilege{
    org.okiproject.serviceapi.roles.GrantPrivilege.GRANTABLEFORGRANTING;
}
java.util.GregorianCalendar effCal =
    new java.util.GregorianCalendar(2001, java.util.Calendar.NOVEMBER, 1);
long effTime = effCal.getTime().getTime();
java.sql.Timestamp effDate = new java.sql.Timestamp(effTime);
java.util.GregorianCalendar expCal =
    new java.util.GregorianCalendar(2001, java.util.Calendar.NOVEMBER, 2);
long expTime = expCal.getTime().getTime();
java.sql.Timestamp expDate = new java.sql.Timestamp(expTime);
try {
    authorization4.update(person4, function4, qualifier4, priv4, TRUE, effDate, expDate);
    System.out.println("updated starting on " + authorization4.getEffectiveDate() + " until " + authorization4.getExpirationDate() + "\nmodified: " + authorization4.getModifiedDate());
    System.out.println();
} catch (Exception ex5) {
    System.out.println(ex5.getMessage());
}

/******************************************************************************
** Get a list of the Functions available within a given category
*******************************************************************************/
try {
    org.okiproject.serviceapi.roles.FunctionCategory category5 =
        myFactory.newFunctionCategory("WRHS");
    System.out.println("List of Functions");
    while (functionList5.hasMoreFunctions()) {
        // Print list of functions
        org.okiproject.serviceapi.roles.Function func =
            functionList5.nextFunction();
        System.out.println("Category: " + category5.getCategory() + ", Function: " + func.getName());
    }
    System.out.println("end list of Functions");
} catch (Exception ex5) {
    System.out.println(ex5.getMessage());
} catch (Exception exx) {
    System.out.println(exx.getMessage());
    exx.printStackTrace();
}
package org.okiproject.serviceapi.usermsg;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute,
 * sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */
public interface UserMessage
extends java.io.Serializable {

/**
 * Method getHeader -- returns the message preamble / headers.
 */
public Object getHeader();

/**
 * Method getBody -- returns the text body.
 */
public Object getBody();

/**
 * Method getTrailer -- returns the trailing headers.
 */
public Object getTrailer();

/**
 * Method getAgent
 */
public org.okiproject.serviceapi.common.Agent getAgent();
}
package org.okiproject.serviceapi.usermsg;

public interface UserMessageQueue extends java.io.Serializable {

    /**
     * Method add
     * @param msg -- The UserMessage
     */
    public void add( UserMessage msg );

    /**
     * Method remove
     * @param msg -- The UserMessage
     */
    public void remove( UserMessage msg );

    /**
     * Method getUserMessageQueueIterator -- returns an iterator.
     */
    public UserMessageQueueIterator getUserMessageQueueIterator();
}
package org.okiproject.serviceapi.usermsg;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 * Purpose: Abstracts the User Message format.
 * Source: /cvs/oki/org/okiproject/serviceapi/usermsg/UserMessageQueueIterator.java,v $@version@ $Revision: 1.1 $$Date: 2002/02/10 23:23:38 $
 */

public interface UserMessageQueueIterator
extends java.io.Serializable {

/**
 * Method next
 *
 * @return UserMessage
 */
   public UserMessage next();

/**
 * Method hasNext();
 *
 */
   public Boolean hasNext();
};
package org.okiproject.serviceapi.usermsg;

import org.okiproject.serviceapi.common.*;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
generated to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
CLAIMS OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

Purpose: Abstracts a User Messaging Service subscriber.
$Source: /cvs/oki/org/okiproject/serviceapi/usermsg/UserMsgClient.java,v $
@version $Revision: 1.4 $
$Date: 2002/02/10 23:22:25 $
 */

public interface UserMsgClient {

/**
 * Method subscribe
 *
 * @param me
 * @param topic
 */
 public void subscribe(final Agent me, final String topic);

/**
 * Method unsubscribe
 *
 * @param me
 * @param topic
 */
 public void unsubscribe(final Agent me, final String topic);

/**
 * Method send
 *
 * @param me
 * @param you
 * @param topic
 * @param msg
 */
 public void send(final Agent me, final Agent you, final String topic,
 final UserMessage msg);

/**
 * Method receive
 *
 * @param me
 * @param you
 * @param topic
 * @param msg
 */
 public void receive(final Agent me, final Agent you, final String topic,
 final UserMessage msg);
}
package org.okiproject.serviceapi.usermsg;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public class UserMsgException extends java.lang.Exception {

    public static final String UNIMPLEMENTED_ERROR = "Not implemented.";
    public static final UserMsgException UNIMPLEMENTED_EXCEPTION = new UserMsgException(UNIMPLEMENTED_ERROR);
    public static final String DEFAULT_FACTORY>Loading_ERROR = "Default factory did not load.";
    public static final UserMsgException DEFAULT_FACTORY_LOADING_EXCEPTION = new UserMsgException(DEFAULT_FACTORY_LOADING_ERROR);
    public static final String PROPERTIES_LOADING_ERROR = "Property file did not load.";
    public static final UserMsgException PROPERTIES_LOADING_EXCEPTION = new UserMsgException(PROPERTIES_LOADING_ERROR);

    /**
     * Constructor UserMsgException
     * 
     */
    public UserMsgException() {
        super();
    }

    /**
     * Constructor UserMsgException
     * 
     * @param message
     */
    public UserMsgException(String message) {
        super(message);
    }
}
org/okiproject/serviceapi/usermsg/UserMsgFactory.java

```java
package org.okiproject.serviceapi.usermsg;

/**
   * Copyright (c) 2001 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software,
and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
 TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

public class UserMsgFactory implements java.io.Serializable {

    static final String USERMSG_FACTORY_NAME = "UserMsgFactory";

    protected static final org.okiproject.serviceapi.usermsg.UserMsgException UNIMPLEMENTED =
            org.okiproject.serviceapi.usermsg.UserMsgException.UNIMPLEMENTED_EXCEPTION;

    private java.util.Map properties;
    private org.okiproject.serviceapi.common.Agent agent;

    /**
     * Method getDefaultUserMsgFactory
     * @param a
     * @param props
     * @throws Exception
     */
    public static org.okiproject.serviceapi.usermsg.UserMsgFactory getDefaultUserMsgFactory(final org.okiproject.serviceapi.common.Agent a, java.util.Properties props) throws Exception {
        if (props == null) {
            throw org.okiproject.serviceapi.usermsg.UserMsgException.PROPERTIES_LOADING_EXCEPTION;
        }
        return getUserMsgFactory(a, props);
    }

    /**
     * Method getUserMsgFactory
     * @param agent
     * @param props
     * @throws Exception
     */
    public static org.okiproject.serviceapi.usermsg.UserMsgFactory getUserMsgFactory(final org.okiproject.serviceapi.common.Agent agent, java.util.Map props) throws Exception {
        String factoryClassName = (String) props.get(USERMSG_FACTORY_NAME);
        if (factoryClassName != null) {
            Class factoryClass = Class.forName(factoryClassName);
            org.okiproject.serviceapi.usermsg.UserMsgFactory factory = (org.okiproject.serviceapi.usermsg.UserMsgFactory) factoryClass.newInstance();
            factory.setProperties(props);
            factory.setAgent(agent);
            return factory;
        }
        throw org.okiproject.serviceapi.usermsg.UserMsgException.DEFAULT_FACTORY_LOADING_EXCEPTION;
    }

    protected static java.util.Properties getPropertiesFromPRB() throws Exception {
        java.util.PropertyResourceBundle prb = (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle(USERMSG_FACTORY_NAME);
        if (prb != null) {
            java.util.Properties props = new java.util.Properties();
            java.util Enumeration keys = prb.getKeys();
            while (keys.hasMoreElements()) {
                String key = (String) keys.nextElement();
                props.setProperty(key, (String) prb.handleGetObject(key));
            }
            return props;
        }
        throw org.okiproject.serviceapi.usermsg.UserMsgException.PROPERTIES_LOADING_EXCEPTION;
    }

```
/**
 * @return Properties.
 */
public java.util.Map getProperties() {
    return this.properties;
}

/**
 * set Properties.
 */
protected void setProperties(java.util.Map properties) {
    this.properties = properties;
}

protected void setAgent(org.okiproject.serviceapi.common.Agent agent) {
    this.agent = agent;
}
package org.okiproject.serviceapi.usermsg;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

Purpose: Abstracts the User Messaging Service which clients can contact
$Source: /cvs/oki/org/okiproject/serviceapi/usermsg/UserMsgService.java,v $
@version $Revision: 1.4 $
$Date: 2002/02/10 23:27:54 $
*/
public interface UserMsgService
extends java.io.Serializable {

/**
 * Method subscribe
 *
 * @param agent
 * @param topic
 */
public void subscribe(org.okiproject.serviceapi.common.Agent agent,
org.okiproject.serviceapi.common.Agent to,
UserMessage msg);

/**
 * Method unsubscribe
 *
 * @param agent
 * @param topic
 */
public void unsubscribe(org.okiproject.serviceapi.common.Agent agent,
String topic);

/**
 * Method send
 *
 * @param from
 * @param to
 * @param msg
 */
public void send(org.okiproject.serviceapi.common.Agent from,
org.okiproject.serviceapi.common.Agent to,
UserMessage msg);

/**
 * Method broadcast
 *
 * @param from
 * @param msg
 */
public void broadcast(org.okiproject.serviceapi.common.Agent from,
UserMessage msg);

```java
```
OKI Common Service Implementations: DISCLAIMER

CAVEAT: The service API implementations in the OKI source tree at org/okiproject/serviceimpl/mit are not intended for use in production. While some implementations have been used, and thus tested, more than others, they have not, in general, been subjected to rigorous testing of any kind, whether unit, regression, system, or integration.

These sample (or reference) implementations serve to illustrate the central OKI philosophy or approach of separating contracts, specifications, or interfaces, expressed as OKI common service APIs, from their implementations, which are replaceable.

A given API could have several implementations, which may be loaded and deployed on demand. Some of the implementations reflect the MIT environment, but are useful, nevertheless.

Please use these implementations as examples illustrating the OKI common services architecture. Use in production is strongly discouraged, and must be at your own risk.

These disclaimers are in addition to any caveats and disclaimers in the license statement and in the source files.
package org.okiproject.serviceimpl.mit.authen;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>
Permission is hereby granted, free of charge, to any<p>person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions):
<main>
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

The Authen API does not contain information that is expected to be part of the implementation. There may be additional implementation information that will need to be exposed in this API.

*/
public class AuthenFactory
    extends org.okiproject.serviceapi.authen.AuthenFactory
{
    private static final org.okiproject.serviceapi.authen.AuthenException UNIMPLEMENDED = new org.okiproject.serviceapi.authen.AuthenException(AuthenException.UNIMPLEMENTED);

    private static final String KERBEROS4 = "KERBEROS 4";

    /**
     * Returns an Agent that may or may not be anonymous
     */
    public org.okiproject.serviceapi.common.Agent getAgent()
        throws Throwable
    {
        return getAgentInternal(null, KERBEROS4);
    }

    /**
     * Returns the types of Authentication that are available to the current implementation
     */
    public String[] getAuthenTypes()
        throws Exception
    {
        String[] types = new String[1];
        types[0] = KERBEROS4;
        return types;
    }

    /**
     * Returns the types of Authentication that this agent currently has.
     */
    public String[] getAuthenTypes( org.okiproject.serviceapi.common.Agent agent)
        throws Exception
    {
        if( agent instanceof org.okiproject.serviceimpl.mit.common.Person )
        {
            org.okiproject.serviceimpl.mit.common.Person person = (org.okiproject.serviceimpl.mit.common.Person) agent;
            String[] types = (String[]) person.getProperty( org.ukiproject.serviceimpl.mit.common.Person.AUTHEN_TYPES );
            return types;
        }
        throw UNIMPLEMENTED;
    }

    public void destroyAuthen( org.okiproject.serviceapi.common.Agent agent)
        throws Exception
    {
        if( agent instanceof org.okiproject.serviceimpl.mit.common.Person )
        {
            org.okiproject.serviceimpl.mit.common.Person person = (org.okiproject.serviceimpl.mit.common.Person) agent;
            person.setProperty( org.ukiproject.serviceimpl.mit.common.Person.AUTHEN_TYPES, null);
        }
        throw UNIMPLEMENTED;
    }

    public void destroyAuthen( org.okiproject.serviceapi.common.Agent agent)
        throws Exception
    {
        if( agent instanceof org.okiproject.serviceimpl.mit.common.Person )
        {
            org.ukiproject.serviceimpl.mit.common.Person person = (org.ukiproject.serviceimpl.mit.common.Person) agent;
            person.setProperty( org.ukiproject.serviceimpl.mit.common.Person.AUTHEN_
public void destroyAuthen
    ( org.okiproject.serviceapi.common.Agent agent,
      String type )
    throws Exception
    {
        if ( ! ( -1 != type.toLowerCase().indexOf( "kerberos" ) ) 
            && ( agent instanceof org.okiproject.serviceimpl.mit.common.Person )
        )
        {
            org.okiproject.serviceimpl.mit.common.Person person
                = ( org.okiproject.serviceimpl.mit.common.Person ) agent;
            person.setProperty( org.okiproject.serviceimpl.mit.common.Person.AUTHEN__
                                , new String[ 0 ] ) ;
            return ;
        }
        throw UNIMPLEMENTED ;
    }

    private boolean isIn( String[] types , String type )
    {
        if ( null != types )
            for ( int i = 0 ; types.length != i ; i ++ )
            {
                if ( -1 != types[ i ].toLowerCase().indexOf( type ) )
                    return true ;
            }
        return false ;
    }

    private org.okiproject.serviceapi.common.Agent getAgentInternal
        ( org.okiproject.serviceapi.common.Agent agent,
          String type )
    { throws Throwable
        {
            javax.swing.JTextField name = new javax.swing.JTextField();
            name.setColumns( 10 ) ;
            javax.swing.JLabel nameLabel = new javax.swing.JLabel( "Name" ) ;
            nameLabel.setLabelFor( name ) ;
            javax.swing.JPasswordField password = new javax.swing.JPasswordField();
            password.setColumns( 10 ) ;
            javax.swing.JLabel passwordLabel = new javax.swing.JLabel( "Password" ) ;
            javax.swing.JLabel quitLabel = new javax.swing.JLabel( "To finish closing dialog" ) ;
            javax.swing.JFrame parent = new javax.swing.JFrame( "Kerberos Authentication" ) ;
            javax.swingInputDialog dialog = new javax.swingInputDialog( parent ) ;
            dialog.setModal( true ) ;
            dialog.getContentPane().setLayout( new java.awt.FlowLayout() ) ;
            dialog.getContentPane().setSize( 100 , 100 ) ;
            dialog.getContentPane().add( nameLabel ) ;
            dialog.getContentPane().add( passwordLabel ) ;
            dialog.getContentPane().add( quitLabel ) ;
            dialog.pack() ;
            dialog.show() ;
            Object ticket = org.okiproject.serviceimpl.mit.krb4.Authenticate.getUserCredentials( name.getText(), 
                                                                                                   "", 
                                                                                                   password.getPassword() ) ;
            String[] types = new String[ 1 ] ;
            types[ 0 ] = KERBEROS4 ;
            if ( null == agent )
                java.util.Properties props = new java.util.Properties() ;
                props.put( org.okiproject.serviceimpl.mit.common.Person.KERBEROS_S_NAME, 
                                                                                                   name.getText() ;
                props.put( org.okiproject.serviceimpl.mit.common.Person.CREDENTIALS_IALS, 
                                                                                                   ticket ;
                props.put( org.okiproject.serviceimpl.mit.common.Person.AUTHEN__
                                , types ) ;
agent = new org.okiproject.serviceimpl.mit.common.Person( props );
} else {

{ org.okiproject.serviceimpl.mit.common.Person agent ).setProperty
   { org.okiproject.serviceimpl.mit.common.Person.KERBERO
      , name.getText() }

{ org.okiproject.serviceimpl.mit.common.Person agent ).setProperty
   { org.okiproject.serviceimpl.mit.common.Person.CREDENTIAL
      , ticket }

{ org.okiproject.serviceimpl.mit.common.Person agent ).setProperty
   { org.okiproject.serviceimpl.mit.common.Person.AUTHENTICATION
      , types }

} return agent ;
}
package org.okiproject.serviceimpl.mit.authencert;
/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of the
Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/
public class AuthenFactory
    extends org.okiproject.serviceapi.authen.AuthenFactory
{
    private static final org.okiproject.serviceapi.authen.AuthenException UNIMPLEMENTED
        = new org.okiproject.serviceapi.authen.AuthenException
            ( org.okiproject.serviceapi.authen.AuthenException.UNIMPLEMENTED ) ;

    private static final String SSL_CERTIFICATE = "SSL CERTIFICATE" ;
    /**
     * Returns an Agent that may or may not be anonymous
     */
    public org.okiproject.serviceapi.common.Agent getAgent()
        throws Throwable
    {
        return getAgentInternal( null , SSL_CERTIFICATE ) ;
    }
    /**
     * Returns the types of Authentication that are available
to the current implementation
     */
    public String[] getAuthenTypes()
    {
        String[] types = new String[ 1 ] ;
        types[ 0 ] = SSL_CERTIFICATE ;
        return types ;
    }
    /**
     * Returns the types of Authentication that this
     * agent currently has.
     */
    public String[] getAuthenTypes
        ( org.okiproject.serviceapi.common.Agent agent )
        throws Exception
    {
        if( agent instanceof org.okiproject.serviceimpl.mit.common.Person )
        {
            org.okiproject.serviceimpl.mit.common.Person person
                = { org.okiproject.serviceimpl.mit.common.Person )
                .AUTHEN_TYPES ) ;
            return types ;
        }
        throw UNIMPLEMENTED ;
    }
    public org.okiproject.serviceapi.common.Agent getAgent
        ( org.okiproject.serviceapi.common.Agent agent , String type )
        throws Throwable
    {
        System.out.println( "getAgent" ) ;
        System.out.println( agent ) ;
        System.out.println( type ) ;

        if( (type != null)
            && ( -1 != type.toLowerCase().indexOf( "ssl certificate" ) )
            && ( agent instanceof org.okiproject.serviceimpl.mit.common.Person ) )
        {
            return getAgentInternal( agent , SSL_CERTIFICATE ) ;
        }
        throw UNIMPLEMENTED ;
    }
    public void destroyAuthen
        ( org.okiproject.serviceapi.common.Agent agent )
        throws Exception
    {
        if( (agent instanceof org.okiproject.serviceimpl.mit.common.Person ) )
        {
            org.okiproject.serviceimpl.mit.common.Person person
                = { org.okiproject.serviceimpl.mit.common.Person )
                .AUTHEN_TYPES ) ;
            person.setProperty
        }
    }
}
public void destroyAuthen
    ( org.okiproject.serviceapi.common.Agent agent                , String type                )
    throws Exception
    {
        if
            ( (type != null)
            && ( -1 != type.toLowerCase().indexOf( "ssl certificate" ) )
            && ( agent instanceof org.okiproject.serviceimpl.mit.common.Person ) )
            {
                org.okiproject.serviceimpl.mit.common.Person person
                    = ( org.okiproject.serviceimpl.mit.common.Person ) agent;
                person.setProperty( org.okiproject.serviceimpl.mit.common.Person.AUTHEN_ TYPES
                        , new String[ 0 ]
                    );
                return;
            }
            throw UNIMPLEMENTED;
    }

public void destroyAuthen
    ( org.okiproject.serviceapi.common.Agent agent                , String[] types                )
    throws Exception
    {
        if
            ( isIn( types , "ssl certificate" )
            && ( agent instanceof org.okiproject.serviceimpl.mit.common.Person ) )
            {
                org.okiproject.serviceimpl.mit.common.Person person
                    = ( org.okiproject.serviceimpl.mit.common.Person ) agent;
                person.setProperty( org.okiproject.serviceimpl.mit.common.Person.AUTHEN_ TYPES
                        , new String[ 0 ]
                    );
                return;
            }
            throw UNIMPLEMENTED;
    }

private boolean isIn( String[] types , String type )
{
    if
        ( (null != types) && (null != type) )
        {
            for( int i = 0 ; types.length != i ; i ++ )
            {  
                if
                    ( -1 != types[ i ].toLowerCase().indexOf( type ) )
                    {
                        return true;
                    }
            }
            return false;
        }
}

private org.okiproject.serviceapi.common.Agent getAgentInternal
    ( org.okiproject.serviceapi.common.Agent agent                , String type                )
    throws Throwable
    {
        // get request and response from the thread
        String username   = (String) request.getAttribute("SSL_CLIENT_S_DN_Email");
        String[] types   = new String[ 1 ];
        types[ 0 ] = SSL_CERTIFICATE;
        String kerberos_name;
        if
            (username.indexOf("@") != -1) {
                kerberos_name = username.substring(0,username.indexOf("@"));
            } else {
                kerberos_name = username;
            }
            if
                (null == agent )
                {
                    java.util.Properties props = new java.util.Properties();
                    props.put
                        ( org.okiproject.serviceimpl.mit.common.Person.KERBERO S_NAME
                            , kerberos_name
                        );
                    props.put
                        ( org.okiproject.serviceimpl.mit.common.Person.FIRST_NAME
                            , (String) request.getAttribute("SSL_CLIENT_S_DN_CN"")
                        );
                    props.put
                        ( org.okiproject.serviceimpl.mit.common.Person.LAST_NAME
                            , (String) request.getAttribute("SSL_CLIENT_S_DN")
                        );
                    props.put
                        ( org.okiproject.serviceimpl.mit.common.Person.AUTHEN_ TYPES
                            , types
                        );
                    agent = new org.okiproject.serviceimpl.mit.common.Person( prop s ) ;
                } else {
                    agent = new org.okiproject.serviceimpl.mit.common.Person( prop s ) ;
                }
    }
org/okiproject/serviceimpl/mit/authencert/AuthenFactory.java

    ( { org.okiproject.serviceimpl.mit.common.Person } agent ).set
  
  S_NAME
  { org.okiproject.serviceimpl.mit.common.Person.KERBERO
  , username
  } ;
  
  Property
  { org.okiproject.serviceimpl.mit.common.Person agent }.set
  TYPES
  { org.okiproject.serviceimpl.mit.common.Person.AUTHEN_
  , types
  } ;

  return agent ;

}
package org.okiproject.serviceimpl.mit.common;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology</p>
<p>Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
</p>
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
</p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
</p>
$Source: /cvs/oki/org/okiproject/serviceimpl/mit/common/Person.java,v $
$Revision: 1.5 $
$Date: 2002/01/25 13:35:42 $

*/

public class Person
    implements org.okiproject.serviceapi.common.Person
{
    private java.util.Properties properties = null;
    public java.util.Properties getProperties() { return this.properties; }
    public void setProperties(java.util.Properties properties) { this.properties = properties; }
    private void setProperty(String propertyName, Object property) {
        if (null != this.properties) {
            this.properties.put(propertyName, property);
        }
    }

    public Object getID() { return getKerberosName(); }
    public void setID(Object id) {
        setKerberosName((String) id);
    }

    public String getName() {
        String name = null;
        if (null != getFirstName()) {
            name = getFirstName();
        }
        if (null != getLastName()) {
            if (null != name) {
                name += " ";
                name += getLastName();
            } else {
                name = getLastName();
            }
        }
        if (null == name) {
            name = getKerberosName();
        }
        return name;
    }

    public String getKerberosName() {
        return (String) getProperty(KERBEROS_NAME);
    }
    protected void setKerberosName(String kerberosName) {
        setProperty(KERBEROS_NAME, kerberosName.toUpperCase());
    }

    public String getMiddleName() {
        return (String) getProperty(MIDDLE_NAME);
    }
    protected void setMiddleName(String middleName) {
        setProperty(MIDDLE_NAME, middleName.toUpperCase());
    }

    public String getSSN() {
        return (String) getProperty(SSN);
    }
    protected void setSSN(String ssn) {
        setProperty(SSN, ssn);
    }

    public static String CREDENTIALS = "CREDENTIALS";
    public static String AUTHEN_TYPES = "AUTHEN_TYPES";
    public static String KERBEROS_NAME = "KERBEROS_NAME";
    public static String MIT_ID = "MIT_ID";
    public static String FIRST_NAME = "FIRST_NAME";
    public static String MIDDLE_NAME = "MIDDLE_NAME";
    public static String LAST_NAME = "LAST_NAME";
    public static String SSN = "SSN";
    public static String DOB = "DOB";
}
public String getDOB()
{
    return (String) getProperty(DOB);
}

protected void setDOB(String dob)
{
    setProperty(DOB, dob);
}

public String getMitid()
{
    return (String) getProperty/MIT_ID);
}

protected void setMitid(String mitid)
{
    setProperty/MIT_ID, mitid);
}

public String getLastName()
{
    return (String) getProperty(LAST_NAME);
}

protected void setLastName(String lastName)
{
    setProperty(LAST_NAME, lastName);
}

public String getFirstName()
{
    return (String) getProperty(FIRST_NAME);
}

protected void setFirstName(String firstName)
{
    setProperty(FIRST_NAME, firstName);
}

public String[] getAuthenTypes()
{
    return (String[]) getProperty(AUTHEN_TYPES);
}

protected void setAuthenTypes(String[] authenTypes)
{
    setProperty(AUTHEN_TYPES, authenTypes);
}

public byte[] getCredentials()
{
    return (byte[]) getProperty(CREDENTIALS);
}

protected void setCredentials(byte[] credentials)
{
    setProperty(CREDENTIALS, credentials);
}

public Person(java.util.Properties properties)
{
    if (null != properties)
    {
        setProperties(new java.util.Properties());
    }
}

public Person(String kerberosName, String mitid, String firstName, String middleName, String lastName, String ssn, String dob)
{
    setProperties(new java.util.Properties());
    setKerberosName(kerberosName);
    setMitid(mitid);
    setFirstName(firstName);
    setMiddleName(middleName);
    setLastName(lastName);
    setSSN(ssn);
    setDOB(dob);
}

public static org.okiproject.serviceimpl.mit.common.Person create(String kerberosName)
{
    return new org.okiproject.serviceimpl.mit.common.Person(kerberosName, null, null, null, null, null, null, null);
}
package org.okiproject.serviceimpl.mit.guid;

/**<p>                                         Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p>
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.<p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.<p>
***********************************************************************/

import java.sql.*;

public interface IDBMSConnectionFactory
{
    public Connection getConnection( String propFile ) throws Exception;
    public Connection getConnection( java.util.Properties props ) throws Exception;
    public Connection getConnection( java.util.PropertyResourceBundle prb) throws Exception;
}
package org.okiproject.serviceimpl.mit.guid;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software,
and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p>
The above copyright notice and this permission notice shall be included in all copies or substantial portions of
the Software.
</p>
The SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED
TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
</p>

* Purpose: Quick-and-dirty Connection factory implementation.
* $Source: /cvs/okiproject/serviceimpl/mit/guid/IDBMSConnectionFactoryImpl.jav
a, revision 1.3 $
* $Date: 2001/12/07 00:07:26 $
* $Revision: 1.3 $
* $Author: <B>Giri Swaminathan, MIT</B>

public class IDBMSConnectionFactoryImpl
implements org.okiproject.serviceimpl.mit.guid.IDBMSConnectionFactory
{
    public java.sql.Connection getConnection( String propFile )
        throws Exception {
        java.io.InputStream is = new java.io.FileInputStream(propFile);
        java.util.Properties p = new java.util.Properties();
p.load( is );
        return getConnection( p );
    }

    public java.sql.Connection getConnection( final java.util.Properties props
        )
        throws Exception {
        String dbDriver = null;
        String dbSpec = null;
        String dbName = null;
        String dbUser = null;
        String dbPass = null;
        if ( (dbDriver = props.getProperty("org.okiproject.serviceimpl.mit.guid.dbDriver")) == null )
            throw new Exception("Property org.okiproject.serviceimpl.mit.guid.dbDriver undefined!
            Aborting . . .");
        try {
            Class c = Class.forName("org.okiproject.serviceimpl.mit.guid.dbDriver");
            c.newInstance();
        } catch (ClassCastException cne) {
            System.out.println("Cannot load database driv

package org.okiproject.serviceimpl.mit.guid;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p>The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.<p>*******************************************************************************/

//import org.okiproject.serviceapi.sql.*;
//import org.okiproject.serviceimpl.mit.sql.*;
public class MitGUID implements org.okiproject.serviceapi.guid.GUID,
java.io.Serializable {

private long id = 0;
private final String procSQL = "(\? = call GET_NEXT_INSTANCE (\?) )";
private final String addSQL = "(\? = call ADD_GUID_TYPE (\?) )";
private String propFile = null;
private String guidType = null;
private java.sql.Connection c = null;

private static final void Usage() {
    System.out.println("Usage: MitGUID <propsFile> <guidType>");
}

public static void main( final String[] args ) {
    int ac = args.length;
    if ( ac < 2 ) {
        Usage();
        System.exit( 1 );
    }
    try {
        final String propsFile = args[ 0 ];
        final String guidType = args[ 1 ];
        MitGUID mitGUIDFactory = new MitGUID( propsFile, guidType );
        Object guid = mitGUIDFactory.getGUID( null, guidType );
        System.out.println("Got guid = "+ guid + " for guidType "+ guidType);
    } catch( java.sql.SQLException ex ) {
        System.out.println("\n--- SQLException caught ---\n");
        while (ex != null) {
            System.out.println("Message: "+ ex.getMessage ());
            System.out.println("SQLState: "+ ex.getSQLState ());
            System.out.println("ErrorCode: "+ ex.getErrorCode ());
            ex = ex.getNextException();
        }
    } catch (final Exception genex ) {
        genex.printStackTrace();
    }
}

public MitGUID( final String propFile, final String guidType ) throws Exception {
    this.propFile = propFile;
    this.guidType = guidType;
}

/** * This will create a GUID for a given kind of OKI entity * @param Agent Caller ID * @param String prefix or "type" of GUID * */
public void addType( final org.okiproject.serviceapi.common.Agent agent,
final String type ) throws org.okiproject.serviceapi.guid.GUIDException {
    /* Ignores return value */
    execGUIDfunc( addSQL, type );
}

private final Long execGUIDfunc( final String SQL, final String type ) throws org.okiproject.serviceapi.guid.GUIDException {
    java.sql.Connection c = null;
    java.sql.CallableStatement cs = null;
    try {
        c = new org.okiproject.serviceimpl.mit.guid.IDBMSConnectionFactory().getConnection( propFile );
        cs = c.prepareCall( SQL );
        System.out.println("\n--- PreparedStatement created ---\n");
        cs.setString( 1, type );
        cs.executeUpdate();
        System.out.println("\n--- PreparedStatement executed ---\n");
        return (Long) cs.getObject( 1 );
    } catch( java.sql.SQLException ex ) {
        System.out.println("\n--- SQLException caught ---\n");
        while (ex != null) {
            System.out.println("Message: "+ ex.getMessage ());
            System.out.println("SQLState: "+ ex.getSQLState ());
            System.out.println("ErrorCode: "+ ex.getErrorCode ());
            ex = ex.getNextException();
        }
    } catch (final Exception genex ) {
        genex.printStackTrace();
    }
    return null;
}
private final Long getIdForType(String type) throws org.okiproject.serviceapi.guid.GUIDException {
    java.sql.Connection c = null;
    java.sql.CallableStatement cs = null;
    try {
      c = new org.okiproject.serviceimpl.mit.guid.IDBMSConnectionFactoryImpl().getConnection(propFile);
      cs = c.prepareCall(procSQL);
      cs.registerOutParameter(1, java.sql.Types.INTEGER);
      cs.setString(2, type);
      cs.execute();
      int id = cs.getInt(1);
      return new Long(id);
    } catch (java.sql.SQLException ex) {
      System.out.println("Message: " + ex.getMessage());
      System.out.println("SQLState: " + ex.getSQLState());
      System.out.println("ErrorCode: " + ex.getErrorCode());
      ex = ex.getNextException();
      System.out.println();
    } finally {
      try {
        cs.close();
        c.close();
      } catch (Exception ignored) {
        // don't care
      } finally {
        cs = null;
        c = null;
      }
    }
  }
}
package org.okiproject.serviceimpl.mit.krb4;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>   person obtaining a copy of this software and associated<p>   documentation files (the "Software"), to deal in the<p>   Software without restriction, including without limitation<p>   the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>   Software, and to permit persons to whom the Software is<p>   furnished to do so, subject to the following conditions:<p>   The above copyright notice and this permission notice<p>   shall be included in all copies or substantial portions of the<p>   Software.</p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p>   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>   WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>   PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>   DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>   CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.</p>*/

public class Authenticate {
    private static String crealm = "ATHENA.MIT.EDU";
    private static String srealm = "ATHENA.MIT.EDU";
    private static String kdc = "KERBEROS.MIT.EDU";

    private static String sinstFrom( String alias )
        throws Exception{
            java.net.InetAddress aliasAddr = java.net.InetAddress.getByName( alias );
            java.net.InetAddress realAddr = java.net.InetAddress.getByName( aliasAddr.getHostAddress() );
            String realHost = realAddr.getHostName().toLowerCase();
            int index = realHost.indexOf(".");
            if(-1 != index ) {
                realHost = realHost.substring( 0 , index );
            }
            return realHost;
        }

    public static krb4.lib.Krb4Creds getUserCredentials
        ( String clientName, String clientInst,
        char[] clientPassword )
        throws Throwable {
            byte[] key = krb4.lib.crypto.des.string_to_key_bytes( new String( clientPassword ) );
            krb4.lib.Krb4ASReq as_req = new krb4.lib.Krb4ASReq{
                clientName ,
                clientInst,
                crealm,
                srealm };
            as_req.send( kdc );
            krb4.lib.Krb4ASRep as_rep = as_req.getKrb4ASRep();
            return as_rep.getCreds( key );
        }

    public static byte[] getServiceTicket
        ( krb4.lib.Krb4Creds clientCredentials,
        String service,
        String host,
        String clientName,
        String clientInst )
        throws Throwable {
            if( null != clientCredentials ) {
                    clientCredentials ,
                    clientName ,
                    clientInst ,
                    crealm ,
                    service ,
                    sinstFrom( host ) ,
                    srealm ,
                    false };
                tgs_req.send( kdc );
                krb4.lib.Krb4TGSRep tgs_rep = tgs_req.getKrb4TGSRep();
                krb4.lib.Krb4Creds creds = tgs_rep.getCreds( clientCredentials );
                krb4.lib.Krb4APReq ap_req = new krb4.lib.Krb4APReq{
                    creds ,
                    clientName ,
                    clientInst ,
                    crealm ,
                    false };
                return ap_req.encode();
            }
            return null;
        }

    public static String isValidClient
        ( byte[] clientTicket ,
        byte[] clientAddress,
        byte[] srvtab )
        throws Throwable {
            krb4.lib.Krb4Encode ref = new krb4.lib.Krb4Encode( srvtab );
            String service = ref.getString();
            String instance = ref.getString();
            String realm = ref.getString();
            byte[] keyVersion = ref.getByte();
            byte[] servKey = ref.getByte();
            krb4.lib.Krb4APReq ap_req = new krb4.lib.Krb4APReq( clientTicket );
            ap_req.authenticate
        }
}
org/okiproject/serviceimpl/mit/krb4/Authenticate.java

```java
    servKey, service, instance, realm, clientAddress)
    return new String(ap_req.getSessionKey());
```
package org.okiproject.serviceimpl.mit.logging;

/**
   Purpose: Represents the formatting strategy used for the message.
*/
public class SimpleDateFormatterImpl implements org.okiproject.serviceapi.logging.Formatter, java.io.Serializable {
    private final java.text.Format sdf = new java.text.SimpleDateFormat("yyyy-M-d HH:mm:ss.SSS a: ");
    
    public String format(final org.okiproject.serviceapi.common.Agent who, final String message) {
        // if (!who.isAuthenticated()) who.authenticate();
        String dateString = sdf.format(new java.util.Date());
        return dateString + message;
    }
};

Copyright (c) 2001 Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/serviceimpl/mit/logging/SimpleDateFormatterImpl.java,v $
$Revision: 1.1 $
$Date: 2002/02/05 19:24:42 $
package org.okiproject.serviceimpl.mit.rmissl;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

AuthenticateException extends Exception {}
package org.okiproject.serviceimpl.mit.rmisssl;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */

public class Client implements org.okiproject.serviceapi.rmi.Client,
org.okiproject.serviceimpl.mit.rmisssl.Remote {
  private String host = null;

  /**
   * Method getHost
   *
   * @return String
   */
  public String getHost() {
    return this.host;
  }

  protected void setHost(String host) {
    this.host = host;
  }

  private Integer port = null;

  /**
   * Method getPort
   *
   * @return Integer
   */
  public Integer getPort() {
    return this.port;
  }

  protected void setPort(Integer port) {
    this.port = port;
  }

  private String service = null;

  /**
   * Method getService
   *
   * @return String
   */
  public String getService() {
    return this.service;
  }

  protected void setService(String service) {
    this.service = service;
  }

  private String server = null;

  /**
   * Method getServer
   *
   * @return String
   */
  public String getServer() {
    return this.server;
  }

  protected void setServer(String server) {
    this.server = server;
  }

  private org.okiproject.serviceapi.common.Agent forAgent = null;

  /**
   * Method getForAgent
   *
   * @return org.okiproject.serviceapi.common.Agent
   */
  public org.okiproject.serviceapi.common.Agent getForAgent() {
    return this.forAgent;
  }

  protected void setForAgent(org.okiproject.serviceapi.common.Agent forAgent) {
    this.forAgent = forAgent;
  }

  private Object response = null;

  /**
   * Method getResponse
   */
  public Object getResponse() {
  }

  protected void setResponse(Object response) {
  }
}
public Object getResponse() {
    return this.response;
}

protected void setResponse(Object response) {
    this.response = response;
}

private org.okiproject.serviceimpl.mit.rmi.ssl.Remote remote = null;

protected org.okiproject.serviceimpl.mit.rmi.ssl.Remote getRemote() {
    return this.remote;
}

protected void setRemote(
    org.okiproject.serviceimpl.mit.rmi.ssl.Remote remote) {
    this.remote = remote;
}

private Client() {}

/**
 * Method create
 * 
 * @param host
 * @param port
 * @param service
 * @param server
 * @param forAgent
 * 
 * @return org.okiproject.serviceapi.rmi.Client
 * 
 * @throws Throwable
 * 
 */
public static org.okiproject.serviceapi.rmi.Client create(String host, Integer port, String service, String server, org.okiproject.serviceapi.common.Agent forAgent) throws Throwable {
    client.setHost(host);
    client.setPort(port);
    client.setService(service);
    client.setServer(server);
    client.setForAgent(forAgent);
    client.setup();
    return client;
}

/**
 * Method authenticate
 * 
 * @param ticket
 * 
 * @return Boolean
 * 
 * @throws Exception
 * 
 */
public Boolean authenticate(byte[] ticket) throws Exception {
    try {
        if (null != getRemote()) {
            return getRemote().authenticate(ticket);
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    return new Boolean(false);
}

/**
 * Method invoke
 * 
 * @param object
 * @param methodName
 * @param argTypes
 * @param args
 * 
 * @return Object
 * 
 * @throws Exception
 * 
 */
public Object invoke(java.io.Serializable object, String methodName, Class[] argTypes, Object[] args) throws Exception {
    try {
        if (null != getRemote()) {
            Object resp = getRemote().invoke(object, methodName, argTypes, args);
            setResponse(resp);
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    if (getResponse() instanceof Exception) {
        throw (Exception) getResponse();
    }
    return getResponse();
}

/**
 * Method test
 * 
 * @return String
 * 
 * @throws Exception
 * 
 */
public String test() throws Exception {
    try {
        if (null != getRemote()) {
            Object resp = getRemote().test();
            setResponse(resp);
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    if (getResponse() instanceof Exception) {
        throw (Exception) getResponse();
    }
    return getResponse();
}
return "Test failed."
;
protected void setup()
throws Throwable {
setResponse(null);
if (null == System.getSecurityManager()) {
    System.setSecurityManager(new java.rmi.RMISecurityManager());
}
if (null == getRemote()) {
    org.okiproject.serviceimpl.mit.rmissl.Remote remote =
        (org.okiproject.serviceimpl.mit.rmissl.Remote) java.rmi.Naming.lookup("//" + getHost() + ":" + getPort().toString() + "/" + getServer());
setRemote(remote);
if ((null != getForAgent())
    && (getForAgent() instanceof org.okiproject.serviceimpl.mit.common.Person)) {
    org.okiproject.serviceimpl.mit.common.Person person =
        (org.okiproject.serviceimpl.mit.common.Person) getForAgent();
    krb4.lib.Krb4Creds creds =
        (krb4.lib.Krb4Creds) person.getProperty(
            org.okiproject.serviceimpl.mit.common.Person.CREDENTIALS);
    String kerberosName =
        (String) person.getProperty(
            org.okiproject.serviceimpl.mit.common.Person.KERBEROS_NAME);
    if (null != creds) {
        byte[] ticket =
            org.okiproject.serviceimpl.mit.krb4.Authenticate.getServiceTicket(
                creds, getService(), getHost(), kerberosName, "");
    authenticate(ticket);
    }
    }
}
package org.okiproject.serviceimpl.mit.rmissl;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

public class ClientFactory
extends org.okiproject.serviceapi.rmi.ClientFactory
{

/**
   * Method createClient
   *
   * @param host
   * @param port
   * @param service
   * @param server
   * @param forAgent
   *
   * @return org.okiproject.serviceapi.rmi.Client
   *
   * @throws Throwable
   *
   */

public org.okiproject.serviceapi.rmi.Client createClient(String host,
    Integer port, String service, String server,
    org.okiproject.serviceapi.common.Agent forAgent)
throws Throwable {
    return org.okiproject.serviceimpl.mit.rmissl.Client.create(host, port,
        service, server, forAgent);
}
package org.okiproject.serviceimpl.mit.rmissl;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 *
 * Source: /cvs/oki/org/okiproject/serviceimpl/mit/rmissl/ClientSocketFactory.java, v 1.3 $@version$ $Revision: 1.3 $$Date: 2002/02/08 13:45:09 $/*
 * public class ClientSocketFactory
 * implements java.rmi.server.RMIClientSocketFactory, java.io.Serializable {
 * private static final int SOCKET_TIMEOUT = 10000;
 * private static javax.net.SocketFactory sslFactory = null;
 * private static java.rmi.server.RMISocketFactory rmiFactory = null;
 * /
 * Method createSocket
 * *
 * @param host
 * @param port
 * *
 * @return java.net.Socket
 * *
 * @throws java.io.IOException
 * *
 * /public java.net.Socket createSocket(String host, int port)
 * throws java.io.IOException {
 * if (null == sslFactory) {
 * sslFactory = javax.net.ssl.SSLSocketFactory.getDefault();
 * }
 * java.net.SSLSocket s =
 * (java.net.SSLSocket) sslFactory.createSocket(host, port);
 * s.setEnabledCipherSuites(s.getSupportedCipherSuites());
 * s.setSoTimeout(SOCKET_TIMEOUT);
 * return s;
 * }
 */
 * Method equals
 * *
 * @param that
 * *
 * @return boolean
 * *
 * /public boolean equals(Object that) {
 * return (that != null) && (that.getClass() == this.getClass());
 * }
 */
package org.okiproject.serviceimpl.mit.rmissl;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of
 * the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Remote
extends java.rmi.Remote {

/**
 * Method authenticate
 *
 * @param ticket
 *
 * @return Boolean
 *
 * @throws Exception
 */
Boolean authenticate(byte[] ticket)
throws Exception;

/**
 * Method invoke
 *
 * @param obj
 * @param methodName
 * @param types
 * @param args
 *
 * @return Object
 *
 * @throws Exception
 */
Object invoke(java.io.Serializable obj, String methodName, Class[] types, Object[] args)
throws Exception;

/**
 * Method test
 *
 * @return String
 *
 * @throws Exception
 */
String test()
throws Exception;
}
// worry: probably don’t want to require mit.krb4.Authenticate for this

package org.okiproject.serviceimpl.mit.rmissl;

/**
 Copyright (c) 2001 Massachusetts Institute of Technology

 Permission is hereby granted, free of charge, to any
 person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/serviceimpl/mit/rmissl/Server.java,v $   
@version $Revision: 1.5 $   
$Date: 2002/02/08 13:45:09 $   

public class Server
extends java.rmi.activation.Activatable
implements org.okiproject.serviceimpl.mit.rmissl.Remote {
    protected String service = null;
    protected transient byte[] srvtab = null;
    protected transient boolean isAuthenticated = true;
    protected String tag = null;

    /**
     * Constructor Server
     *
     * @param id
     * @param data
     * @throws Exception
     */
    public Server(java.rmi.activation.ActivationID id,
            java.rmi.MarshalledObject data)
        throws Exception {
        super(id, 0,
                (java.rmi.server.RMIClientSocketFactory) new org.okiproject.serviceimpl.mit.rmisl.ClientSocketFactory(),
                (java.rmi.server.RMIServerSocketFactory) new org.okiproject.serviceimpl.mit.rmisl.ServerSocketFactory());
        srvtab = (byte[]) data.get();
        if (null != srvtab) {
            isAuthenticated = false;
        }
    }

    /**
     * Method authenticate
     *
     * @param ticket
     * @return Boolean
     */
    public Boolean authenticate(byte[] ticket) {
        try {
            org.okiproject.serviceimpl.mit.krb4.Authenticate authenticate =
                    new org.okiproject.serviceimpl.mit.krb4.Authenticate();
            byte[] clientAddress =
                java.net.InetAddress.getByName(getClientHost()).getAddress();
            String host =
                java.net.InetAddress.getLocalHost().getHostName();
            this.tag = authenticate.isValidClient(ticket, clientAddress, srvtab);
            org.okiproject.serviceimpl.mit.util.ThreadContext.setProperty("Tag",
                    this.tag);
            isAuthenticated = true;
            return new Boolean(true);
        } catch (Throwable ex) {
            return new Boolean(false);
        }
    }

    /**
     * Method invoke
     *
     * @param object
     * @param methodName
     * @param argTypes
     * @param args
     * @return Object
     */
    public Object invoke(java.io.Serializable object, String methodName,
            Class[] argTypes, Object[] args) {
        Object response = null;
        try {
            if (isAuthenticated) {
                java.lang.reflect.Method method = null;
                if (!object instanceof java.lang.Class) {
                    method = object.getClass().getMethod(methodName, argTypes);
                } else {
                    method = ((Class) object).getMethod(methodName, argTypes);
                }
                // Other code...
```java
    response = method.invoke(object, args);
    }
    else {
        response = new org.okiproject.serviceimpl.mit.rmissl.AuthenticateException();
    }
    }
    catch (Throwable ex) {
        response = ex;
    }
    return response;
}

/**
 * Method test
 *
 * @return String
 */
public String test() {
    return "Test successful";
}
```
package org.okiproject.serviceimpl.mit.rmissl;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
distributed to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DEFENSES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

@see org.okiproject.serviceapi.rmi.ServerFactory

/**
 * $Source: /cvs/oki/org/okiproject/serviceimpl/mit/rmissl/ServerFactory.java,v $
 * $Revision: 1.6 $
 * $Date: 2002/02/08 13:45:09 $
 */
public class ServerFactory
    extends org.okiproject.serviceapi.rmi.ServerFactory {

    /**
     * Method setup
     *
     * @param props
     * @throws Exception
     */
    public void setup(java.util.Properties props)
        throws Exception {
        java.io.FileInputStream in =
            new java.io.FileInputStream((String) props.get("srvtab"));
        byte[] srvtab = new byte[0];
        int length = 0;
        int available = 0;
        while (0 != in.available()) {
            available = in.available();
            byte[] newsrvtab = new byte[length + available];
            System.arraycopy(srvtab, 0, newsrvtab, 0, length);
            in.read(newsrvtab, length, available);
            srvtab = newsrvtab;
            length += available;
        }
        in.close();

        java.util.Properties props2 = new java.util.Properties();
        props2.put(props.get("policyProp"), props.get("policy"));
        java.rmi.activation.ActivationGroupDesc.cmdEnv = null;
        java.rmi.activation.ActivationGroupDesc group =
            new java.rmi.activation.ActivationGroupDesc(props2, ace);
        java.rmi.activation.ActivationGroupID agi =
            java.rmi.activation.ActivationGroupID.getSystem().registerGroup(group);
        java.rmi.activation.ActivationGroupActivationDesc desc =
            new java.rmi.activation.ActivationDesc((String) props.get("server"),
            (String) props.get("location"), data);
        java.rmi.Remote mri =
            java.rmi.activation.Activatable.register(desc);
        java.rmi.Naming.rebind((String) props.get("server"), mri);
        System.out.println("Exported Server");
    }
}
if (null == sslFactory) {
sslFactory = javax.net.ssl.SSLServerSocketFactory.getDefault();
}

if (ANONYMOUS_PORT == ssport) {
ssport = 8000;
}

/**
* Method createServerSocket
*
*
* @param port
*
* @return java.net.ServerSocket
*
* @throws java.io.IOException
*
*/
public java.net.ServerSocket createServerSocket(int port)
throws java.io.IOException {
int
ssport
= port;
Exception exception = null;

<p>
$Source: /cvs/oki/org/okiproject/serviceimpl/mit/rmissl/ServerSocketFactory.java,v $
@version@ $Revision: 1.2 $
$Date: 2002/02/08 13:45:09 $
*/
public class ServerSocketFactory
implements java.rmi.server.RMIServerSocketFactory, java.io.Serializable {
private static final int
ANONYMOUS_PORT = 0;
private static final int
MAX_PORTS
= 100;
private static javax.net.ServerSocketFactory sslFactory
= null;

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

<p>

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

<p>

Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

<p>

}

return s;

/**
* Method equals
*
*
* @param that
*
* @return boolean
*
*/
public boolean equals(Object that) {
return (that != null) && (that.getClass() == this.getClass());
}

}

break;
}
catch (Exception ex) {
exception = ex;
}

exception = null;

if (null != exception) {
throw new java.io.IOException(exception.toString());
}

}

/**
<p>
s.setEnabledCipherSuites(s.getSupportedCipherSuites());

for (int i = 0; MAX_PORTS != i; i++) {
try {
s = (javax.net.ssl.SSLServerSocket) sslFactory.createServerSocket(
ssport + i);

Copyright (c) 2001 Massachusetts Institute of Technology

javax.net.ssl.SSLServerSocket s = null;

org/okiproject/serviceimpl/mit/rmissl/ServerSocketFactory.java

package org.okiproject.serviceimpl.mit.rmissl;

02/11/02
11:42:48

1


package org.okiproject.serviceimpl.mit.sql;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>Permission is hereby granted, free of charge, to any<p>person obtaining a copy of this software and associated<p>documentation files (the "Software"), to deal in the<p>Software without restriction, including without limitation<p>the rights to use, copy, modify, merge, publish, distribute,<p sublicense, and/or sell copies of the<p>Software, and to permit persons to whom the Software is<p>furnished to do so, subject to the following conditions:<p>The above copyright notice and this permission notice<p>shall be included in all copies or substantial portions of<p>the Software.</p></p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p_KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p_PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>IN THE SOFTWARE.</p>*/

public class Blob
    implements org.okiproject.serviceapi.sql.Blob
{
    private static final String GETBYTES = "getBytes";
    private static final String POSITION = "position";
    private static final Long INVALIDPOSITION = new Long( -1 ) ;
    private static final org.okiproject.serviceapi.sql.TableException LOCALBLOBGET_BYTESFAILED =
        new org.okiproject.serviceapi.sql.TableException( org.okiproject.serviceapi.sql.TableException.BLOBGETBYTESFAILED ) ;
    private static final org.okiproject.serviceapi.sql.TableException LOCALBLOBPOSITIONFAILED =
        new org.okiproject.serviceapi.sql.TableException( org.okiproject.serviceapi.sql.TableException.BLOBPOSITIONFAILED ) ;

    private org.okiproject.serviceimpl.mit.sql.Rows rows = null ;
    private int row = 0 ;
    private int col = 0 ;
    private Long pos = INVALIDPOSITION;
    private Integer len = new Integer( 0 ) ;
    private byte[] pattern = null ;
    private Long start = INVALIDPOSITION ;

    protected Blob()
    {
        this.rows = rows ;
        this.row = row ;
        this.col = col ;
        this.pos = pos ;
        this.len = len ;
    }

    public byte[] getBytes( Long pos, Integer len )
        throws Exception
    {
        if( connection.hasClient() )
        {
            bytes = getBytesRemotely() ;
        } else
        {
            bytes = getBytesLocally() ;
        }
        return bytes ;
    }

    private byte[] getBytesRemotely()
        throws Exception
    {
        Object[] args = new Object[ 2 ] ;
        args[ 0 ] = this.pos ;
        args[ 1 ] = this.len ;
        Class[] argstypes = new Class[ 2 ] ;
        argstypes[ 0 ] = args[ 0 ].getClass() ;
        argstypes[ 1 ] = args[ 1 ].getClass() ;
        org.okiproject.serviceimpl.mit.sql.Connection connection =
            ( org.okiproject.serviceimpl.mit.sql.Connection )
                rows.table.getConnection() ;
        return (byte[] ) connection.invoke( this, GETBYTES, argstypes, args ) ;
    }

    public byte[] getBytesLocally()
        throws Exception
    {
        java.sql.ResultSet rs = null ;
        try
        {
            rs = rows.getResultSet() ;
            int rsType = rs.getType() ;
            if( rs.TYPE_FORWARD_ONLY == rsType )
            {
                if( rs.next() )
                {

```
return getBytesFromResultSet(rs);
}

else if(rs.TYPE_SCROLL_INSENSITIVE == rsType)
{
    rs.setFetchDirection(rs.FETCH_FORWARD);
    if(rs.first())
    {
        return getBytesFromResultSet(rs);
    }
}
else if(rs.TYPE_SCROLL_SENSITIVE == rsType)
{
    rs.setFetchDirection(rs.FETCH_FORWARD);
    boolean isRowAvailable = rs.first();
    if(isRowAvailable)
    {
        return getBytesFromResultSet(rs);
    }
}
java.sql.Statement statement = rs.getStatement();
rs.close();
statement.close();
}
catch(Exception ex)
{
    if(null != rs)
    {
        rs.close();
    }
    String message = LOCALBLOBGETBYTESFAILED.getMessage();
    message += ex.getMessage();
    throw new org.okiproject.serviceapi.sql.TableException(message);
}
return null;

private byte[] getBytesFromResultSet(java.sql.ResultSet rs)
throws Exception
{
    int cursor = 0;
    boolean isRowAvailable = true;
    while(isRowAvailable)
    {
        if(cursor == this.row)
        {
            java.sql.Blob blob = (java.sql.Blob) rs.getObject(this.col+1);
            return blob.getBytes(this.pos.longValue(), this.len.intValue());
        }
        cursor ++;
        isRowAvailable = rs.next();
    }
    return null;
}

public Long position(byte[] pattern, Long start)
throws Exception
    return positionFromResultSet(rs);
    }
    java.sql.Statement statement = rs.getStatement();
    rs.close();
    statement.close();
}
    catch(Exception ex)
    {
        if(null != rs)
        {
            rs.close();
        }
        String message = LOCALBLOBPOSITIONFAILED.getMessage();
        message += ex.getMessage();
        throw new org.okiproject.serviceapi.sql.TableException(message);
    }
    return INVALIDPOSITION;

private Long positionFromResultSet(java.sql.ResultSet rs)
throws Exception
{
    int cursor = 0;
    boolean isRowAvailable = true;
    while(isRowAvailable)
    {
        if(cursor == this.row)
        {
            java.sql.Blob blob = (java.sql.Blob) rs.getObject(this.col + 1);
            return new Long(blob.position(this.pattern , this.start.longValue()));
        }
        cursor++;
        isRowAvailable = rs.next();
    }
    return INVALIDPOSITION;
}
package org.okiproject.serviceimpl.mit.sql;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>   person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the<p>   Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>   Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p>   The above copyright notice and this permission notice shall be included in all copies or substantial portions of the<p>   Software. </p>
<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>IN THE SOFTWARE.</p></*/

public class Clob implements org.okiproject.serviceapi.sql.Clob {

    private final String GETSUBSTRING = "getSubString";
    private final String POSITION = "position";
    private final Long INVALIDPOSITION = new Long(-1);

    private static final String LOCALCLOBGETSUBSTRINGFAILED =
            new org.okiproject.serviceapi.sql.TableException.
                    LOCALCLOBGETSUBSTRINGFAILED (this.rows.table.getConnection(),
                    this.pos, this.len);

    private static final String LOCALCLOBPOSITIONFAILED =
            new org.okiproject.serviceapi.sql.TableException.
                    LOCALCLOBPOSITIONFAILED (this.rows.table.getConnection(),
                    this.pos, this.col);

    private final org.okiproject.serviceimpl.mit.sql.Rows rows = null;
    private int row = 0;
    private int col = 0;
    private Long pos = INVALIDPOSITION;
    private Integer len = new Integer(0);
    private String searchstr = null;
    private Long start = INVALIDPOSITION;
    private long length = 0;

    public long length() {
        return this.length;
    }

    public String getSubString(Long pos, Integer len) throws Exception {
        this.pos = pos;
        this.len = len;
        return getSubStringLocally();
    }

    private String getSubStringLocally() throws Exception {
        java.sql.ResultSet rs = null;
        try {
            rs = rows.getResultSet();
            if (rs.next()) {
                int row = rs.getInt(0);
                if (rs.next()) {
                    return this.length;
                }
            }
        } finally {
            rows.close();
        }
        return getSubStringRemotely();
    }

    private String getSubStringRemotely() throws Exception {
        Object[] args = new Object[2];
        args[0] = this.pos;
        args[1] = this.len;
        Class[] argstypes = new Class[2];
        argstypes[0] = args[0].getClass();
        argstypes[1] = args[1].getClass();

        org.okiproject.serviceimpl.mit.sql.Connection connection =
                this.rows.table.getConnection();
        org.okiproject.serviceimpl.mit.sql.Client client =
                connection.getClient();
        String substr = null;
        if (connection.hasClient()) {
            substr = getSubStringRemotely();
        } else {
            substr = getSubStringLocally();
        }
        return substr;
    }

    public String getSubStringLocally() throws Exception {
        try {
            java.sql.ResultSet rs = null;
            try {
                rs = rows.getResultSet();
                int rsType = rs.getType();
                if (rsType == java.sql.ResultSet.TYPE_FORWARD_ONLY) {
                    if (rs.next()) {
                        this.length = 0;
                        return this.length;
                    }
                }
            } finally {
                rows.close();
            }
        }
        return getSubStringRemotely();
    }

}
private String getSubStringFromResultSet(java.sql.ResultSet rs) throws Exception {
    return rs.getString(this.col + 1);
}

private Long positionRemotely() throws Exception {
    Object[] args = new Object[2];
    args[0] = this.searchstr;
    args[1] = this.start;
    Class[] argstypes = new Class[2];
    argstypes[0] = args[0].getClass();
    argstypes[1] = args[1].getClass();
    return (Long) connection.invokeProvider(POSITION, argstypes, args);
}

private Long positionLocally() throws Exception {
    java.sql.ResultSet rs = null;
    try {
        rs = rows.getResultSet();
        int rsType = rs.getType();
        if (rs.TYPE_FORWARD_ONLY == rsType) {
            if (rs.next()) {
                return positionFromResultSet(rs);
            }
        } else if (rs.TYPE_SCROLL_INSENSITIVE == rsType) {
            rs.setFetchDirection(rs.FETCH_FORWARD);
            if (rs.first()) {
                return positionFromResultSet(rs);
            }
        } else if (rs.TYPE_SCROLL_SENSITIVE == rsType) {
            rs.setFetchDirection(rs.FETCH_FORWARD);
            boolean isRowAvailable = rs.first();
            if (isRowAvailable) {
                return positionFromResultSet(rs);
            }
        }
        return null;
    }
    catch (Exception ex) {
        String message = LOCALCLOBGETSUBSTRINGFAILED.getMessage();
        message += ex.getMessage();
        throw new org.okiproject.serviceapi.sql.TableException(message);
    }
    return null;
}

public Long position(String searchstr, Long start) throws Exception {
    this.searchstr = searchstr;
    this.start = start;
    Long thePosition = INVALIDPOSITION;
    if (connection.hasClient()) {
        thePosition = positionRemotely();
    } else {
        thePosition = positionLocally();
    }
    return thePosition;
}
```java
    return positionFromResultSet(rs);
}

java.sql.Statement statement = rs.getStatement();
rs.close();
statement.close();
}

catch(Exception ex)
{
    if(null != rs)
    {
        rs.close();
    }
    String message = LOCALCLOBPOSITIONFAILED.getMessage();
    message += ex.getMessage();
    throw new org.okiproject.serviceapi.sql.TableException(message);
}

return INVALIDPOSITION;

private Long positionFromResultSet(java.sql.ResultSet rs)
throws Exception{
    int cursor = 0;
    boolean isRowAvailable = true;
    while(isRowAvailable)
    {
        if(cursor == this.row)
        {
            java.sql.Clob clob = (java.sql.Clob) rs.getObject(this.col + 1);
            return new Long(clob.position(this.searchstr, this.start.longValue()));
        }
        cursor ++;
        isRowAvailable = rs.next();
    }
    return INVALIDPOSITION;
}
```
protected String schemaName = null;
public String getSchemaName() { return this.schemaName; }

protected String tableName = null;
public String getTableName() { return this.tableName; }

protected boolean autoIncrement = false;
public boolean isAutoIncrement() { return this.autoIncrement; }

protected boolean caseSensitive = false;
public boolean isCaseSensitive() { return this.caseSensitive; }

protected boolean currency = false;
public boolean isCurrency() { return this.currency; }

protected boolean definitelyWritable = false;
public boolean isDefinitelyWritable() { return this.definitelyWritable; }

protected int nullable = 0;
public int getNullable() { return this.nullable; }

protected boolean readOnly = false;
public boolean isReadOnly() { return this.readOnly; }

protected boolean searchable = false;
public boolean isSearchable() { return this.searchable; }

protected boolean signed = false;
public boolean isSigned() { return this.signed; }

protected boolean writable = false;
public boolean isWritable() { return this.writable; }

protected ColumnMetaData implements org.okiproject.serviceapi.sql.ColumnMetaData {
    int column;
    java.sql.ResultSetMetaData rsMetaData;

    public ColumnMetaData {
        this.column = column;
        this.rsMetaData = rsMetaData;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public int getColumn() {
        return this.column;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.catalogName;
    }

    public String getCatalogName() {
        return this.c
package org.okiproject.serviceimpl.mit.sql;

/**<p> Copyright (c) 2001 Massachusetts Institute of Technology<p> Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p> The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.<p> THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.</p>

@version@ @Revision: 1.1 @
@Date: 2001/11/29 00:46:11 @

public class Connect
extends java.lang.Thread
implements java.lang.Cloneable
{
    private static final int MAXRETRIES = 3 ;
    protected String databaseURL = null ;
    protected String username = null ;
    protected String password = null ;
    protected java.sql.Connection conn = null ;

    public Connect(String driverName, String databaseURL, String username, String password)
        throws Exception
    {
        Class.forName( driverName ) ;
        this.databaseURL = databaseURL ;
        this.username = username ;
        this.password = password ;
    }

    public java.sql.Connection getConnection()
    {
        try{
            if( null == conn )
            {
                for( int i = 0 ; i < MAXRETRIES; i++ )
                {
                    clone = (org.okiproject.serviceimpl.mit.sql.Connect) this.clone() ;
                    clone.start() ;
                    clone.join( 30000 ) ; // wait 30 seconds for a timeout
                    conn = clone.conn ;
                    if( null != conn )
                    {
                        return conn ;
                    }
                }
            }
            return conn ;
        }
        catch( Exception ex )
        {
            ex.printStackTrace() ;
            return null ;
        }

    }

    static int count = 0 ;
    public void run()
    {
        try{
            count ++ ;
            conn = java.sql.DriverManager.getConnection( databaseURL , username , password ) ;
        }
        catch( Exception ex )
        {
            ex.printStackTrace() ;
        }
    }
package org.okiproject.serviceimpl.mit.sql;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute,
 sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$/Source:/cvs/oki/org/okiproject/serviceimpl/mit/sql/Connection.java,v $ @version$ $Revision: 1.1 $$Date: 2001/11/29 00:46:11 $*/

public class Connection

implements org.okiproject.serviceapi.sql.Connection
{

private static final String EXECUTEUPDATE = "executeUpdate";
private static final String CALLPROCEDURE = "callProcedure";
private static final String COLUMN_NAME = "COLUMN_NAME";
private static final String COLUMN_TYPE = "COLUMN_TYPE";
private static final String DATA_TYPE = "DATA_TYPE";
private static final String SEQUENCE = "SEQUENCE";

public org.okiproject.serviceapi.sql.Table getProcedures
(String catalog
 , String schemaPattern
 , String procedureNamePattern
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.getProcedures
( this
 , catalog
 , schemaPattern
 , procedureNamePattern
 , columnNamePattern
);
}

public org.okiproject.serviceapi.sql.Table getTables()
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.getTables
( this
);
}

public org.okiproject.serviceapi.sql.Table getTables
(String catalog
 , String schemaPattern
 , String tableNamePattern
 , String[] tableTypes
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.getTables
( this
 , catalog
 , schemaPattern
 , tableNamePattern
 , tableTypes
);
}

public org.okiproject.serviceapi.sql.Table getTables
(String catalog
 , String schemaPattern
 , String tableNamePattern
 , String[] tableTypes
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.getTables
( this
 , catalog
 , schemaPattern
 , tableNamePattern
 , tableTypes
);
}

public org.okiproject.serviceapi.sql.Table getProcedureColumns
(String catalog
 , String schemaPattern
 , String procedureName
 , String columnNamePattern
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.getProcedureColumns
( this
 , catalog
 , schemaPattern
 , procedureName
 , columnNamePattern
);
}

public org.okiproject.serviceapi.sql.Table getProcedureColumns
(String catalog
 , String schemaPattern
 , String procedureName
 , String columnNamePattern
 , String[] targetSqlTypes
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.getProcedureColumns
( this
 , catalog
 , schemaPattern
 , procedureName
 , columnNamePattern
 , targetSqlTypes
);

public org.okiproject.serviceapi.sql.Table executeQuery
(String sql
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.executeQuery
( this
 , sql
);
}

public org.okiproject.serviceapi.sql.Table executeQuery
(String sql
 , Object[] objs
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.executeQuery
( this
 , sql
 , objs
);
}

public org.okiproject.serviceapi.sql.Table executeQuery
(String sql
 , Object[] objs
 , int[] targetSqlTypes
)
throws Exception
{
return org.okiproject.serviceimpl.mit.sql.Table.executeQuery
( this
 , sql
 , objs
 , targetSqlTypes
);
}
public org.okiproject.serviceimpl.mit.sql.Table executeQuery
(String sql
, Object[] objs
, int[] targetSqlTypes
, int[] scales)
throws Exception
{
    return org.okiproject.serviceimpl.mit.sql.Table.executeQuery
    (this
, sql
, objs
, targetSqlTypes
, scales);
}

public int executeUpdate(String sql)
throws Exception
{
    if (hasClient())
    {
        return executeUpdateRemotely(sql);
    }
    return executeUpdateLocally(sql);
}

private int executeUpdateRemotely(String sql)
throws Exception
{
    Class[] argsTypes = new Class[1];
    Object[] args = new Object[1];
    argsTypes[0] = sql.getClass();
    args[0] = sql;
    Number num = (Number)getClient().invoke(this, EXECUTEUPDATE, argsTypes, args);
    return num.intValue();
}

public boolean hasClient()
throws Exception
{
    return org.okiproject.serviceimpl.mit.util.RuntimeContext.contains(new String(clientstr));
}

public org.okiproject.serviceimpl.mit.rmi.Client getClient()
throws Exception
{
    if (hasClient())
    {
    }
    return null;
}

public int executeUpdateLocally(String sql)
throws Exception
{
    java.sql.Statement statement = null;
    try
    {
        java.sql.Connection conn = getConnection();
        statement = conn.createStatement();
        int rowCount = statement.executeUpdate(sql);
        statement.close();
        statement = null;
        return rowCount;
    }
    catch(Exception ex)
    {
        if (null != statement)
        {
            statement.close();
        }
        throw ex;
    }
}

public int executeUpdate(String sql, Object[] objs)
throws Exception
{
    if (hasClient())
    {
        return executeUpdateRemotely(sql, objs);
    }
    return executeUpdateLocally(sql, objs);
}

private int executeUpdateRemotely(String sql, Object[] objs)
throws Exception
{
    Class[] argsTypes = new Class[2];
    Object[] args = new Object[2];
    argsTypes[0] = sql.getClass();
    argsTypes[1] = objs.getClass();
    args[0] = sql;
    args[1] = objs;
    Number num = (Number)getClient().invoke(this, EXECUTEUPDATE, argsTypes, args);
    return num.intValue();
}

public int executeUpdateLocally(String sql, Object[] objs)
throws Exception
{
    java.sql.PreparedStatement statement = null;
    try
    {
        java.sql.Connection conn = getConnection();
        statement = conn.prepareStatement(sql);
        for (int i = 0; objs.length != i; i++)
        {
            statement.setObject(i + 1, objs[i]);
        }
        int rowCount = statement.executeUpdate();
        statement.close();
        statement = null;
        return rowCount;
    }
    catch(Exception ex)
    {
        if (null != statement)
        {
            throw ex;
        }
    }
}
public int executeUpdate(String sql, Object[] objs, int[] targetSqlTypes)
    throws Exception
{
    if (hasClient())
    {
        return executeUpdateRemotely(sql, objs, targetSqlTypes);
    }
    return executeUpdateLocally(sql, objs, targetSqlTypes);
}

private int executeUpdateRemotely(String sql, Object[] objs, int[] targetSqlTypes)
    throws Exception
{
    Class[] argsTypes = new Class[3];
    argsTypes[0] = sql.getClass();
    argsTypes[1] = objs.getClass();
    argsTypes[2] = targetSqlTypes.getClass();
    java.sql.PreparedStatement statement = null;
    try
    {
        java.sql.Connection conn = getConnection();
        statement = conn.prepareStatement(sql);
        for (int i = 0; i < objs.length; i++)
        {
            if (targetSqlTypes != null)
            {
                statement.setObject(i + 1, objs[i], targetSqlTypes[i]);
            }
            else
            {
                statement.setObject(i + 1, objs[i]);
            }
        }
        int rowCount = statement.executeUpdate();
        statement.close();
        return rowCount;
    }
    catch (Exception ex)
    {
        if (null != statement)
        {
            statement.close();
        }
        throw ex;
    }
}

public int executeUpdate(String sql, Object[] objs, int[] targetSqlTypes, int[] scales)
    throws Exception
{
    if (hasClient())
    {
        return executeUpdateRemotely(sql, objs, targetSqlTypes, scales);
    }
    return executeUpdateLocally(sql, objs, targetSqlTypes, scales);
}

private int executeUpdateLocally(String sql, Object[] objs, int[] targetSqlTypes, int[] scales)
    throws Exception
{
    java.sql.PreparedStatement statement = null;
    try
    {
        java.sql.Connection conn = getConnection();
        statement = conn.prepareStatement(sql);
        for (int i = 0; i < objs.length; i++)
        {
            if (scales != null)
            {
                statement.setObject(i + 1, objs[i], scales[i]);
            }
            else
            {
                statement.setObject(i + 1, objs[i]);
            }
        }
        int rowCount = statement.executeUpdate();
        statement.close();
        return rowCount;
    }
    catch (Exception ex)
    {
        if (null != statement)
        {
            statement.close();
        }
        throw ex;
    }
}

public void commit()
    throws Exception
{
try {
    if (! getAutoCommit().booleanValue()) {
        getConnection().commit();
    }
} catch (Exception ex) {
    resetConnection();
    throw ex;
}

public void rollback() throws Exception {
    try {
        if (! getAutoCommit().booleanValue()) {
            getConnection().rollback();
        }
    } catch (Exception ex) {
        resetConnection();
        throw ex;
    }
}

public org.okiproject.serviceapi.sql.Table callProcedure(String procedureName, Object[] args, int[] columnTypes, int[] dataTypes) throws Exception {
    return callProcedure((String) null, (String) null, procedureName, (String) null, args, columnTypes, dataTypes);
}

public org.okiproject.serviceapi.sql.Table callProcedure(String catalog, String schemaPattern, String procedureName, String columnNamePattern, Object[] arguments, int[] columnTypes, int[] dataTypes) throws Exception {
    Class[] argsTypes = new Class[7];
    Object[] args = new Object[7];
    argsTypes[0] = (null != catalog) ? catalog.getClass() : null;
    argsTypes[1] = (null != schemaPattern) ? schemaPattern.getClass() : null;
    argsTypes[2] = (null != arguments) ? arguments.getClass() : null;
    argsTypes[3] = (null != columnTypes) ? columnTypes.getClass() : null;
    argsTypes[4] = (null != dataTypes) ? dataTypes.getClass() : null;
    args[0] = catalog;
    args[1] = schemaPattern;
    args[2] = procedureName;
    args[3] = arguments;
    args[4] = columnTypes;
    args[5] = dataTypes;
    return getClient().invoke(this, CALLPROCEDURE, argsTypes, args);
}

public org.okiproject.serviceapi.sql.Table callProcedureLocally(String catalog, String schemaPattern, String procedureName, String columnNamePattern, Object[] arguments, int[] columnTypes, int[] dataTypes) {
    return null;
}

private org.okiproject.serviceapi.sql.Table callProcedureRemotely(String catalog, String schemaPattern, String procedureName, String columnNamePattern, Object[] arguments, int[] columnTypes, int[] dataTypes) throws Exception {
    return null;
}
org/okiproject/serviceimpl/mit/sql/Connection.java

```java
    java.sql.CallableStatement stmt = null;
    try {
        if (null != procedureName) && (null != args) && (null != columnTypes) && (null != dataTypes) {
            stmt = setStatement(catalog,
                               , schemaPattern,
                               , procedureNamePattern,
                               , args,
                               , columnTypes,
                               , DataTypes);
            stmt.execute();
            org.okiproject.serviceapi.sql.Table table = createTable( stmt , columnTypes , DataTypes ) ;
            stmt.close();
            return table ;
        }
        throw new org.okiproject.serviceapi.sql.ConnectionException
            { org.okiproject.serviceapi.sql.ConnectionException.INVALIDARGUMENTS } ;
    }
    catch( Exception ex ) {
        if( null != stmt ) {
            stmt.close();
        }
        resetConnection();
        throw ex ;
    }

    private java.sql.CallableStatement setStatement
    { String catalog
      , String schemaPattern
      , String procedureNamePattern
      , Object[] args
      , int[] columnTypes
      , int[] DataTypes
    } throws Exception {
        String ret = "";
        String params = "";
        int paramCount = 0;
        for( int i = 0 ; columnTypes.length != i ; i ++ ) {
            int columnType = columnTypes[i];
            switch( columnType ) {
                case java.sql.DatabaseMetaData.procedureColumnIn;
                case java.sql.DatabaseMetaData.procedureColumnOut;
                case java.sql.DatabaseMetaData.procedureColumnInOut:
                case java.sql.DatabaseMetaData.procedureColumnResult:
                    if( 0 != paramCount ) { params += ", " ; }
                    params += ?;
                    paramCount ++ ;
                    break ;
                case java.sql.DatabaseMetaData.procedureColumnReturn:
                    ret = ?=
                    break ;
            }
            String sql = "(" + ret + "call " + procedureName + "(" + params + ")")";
            java.sql.CallableStatement stmt = getConnection().prepareCall( sql ) ;
            int index = 0 ;
            int sequence = 1 ;
            for( int i = 0 ; columnTypes.length != i ; i ++ ) {
                int columnType = columnTypes[i];
                int datatype = DataTypes[i];
                Object arg = null;
                if( (null != args) && (args.length != index)) {
                    arg = args[index++];
                }
                if( null == arg ) {
                    arg = "";
                }
                switch( columnType ) {
                    case java.sql.DatabaseMetaData.procedureColumnIn:
                        if( datatype == java.sql.Types.OTHER ) {
                            stmt.setObject( sequence , arg , java.sql.Types.FLOAT , 0 );
                        } else {
                            stmt.setObject( sequence , arg , datatype );
                        }
                        sequence ++ ;
                        break ;
                    case java.sql.DatabaseMetaData.procedureColumnOut:
                        if( datatype == java.sql.Types.OTHER ) {
                            stmt.setObject( sequence , arg , java.sql.Types.FLOAT , 0 );
                        } else {
                            stmt.registerOutParameter( sequence , java.sql.Types.FLOAT , 0 );
                        }
                        sequence ++ ;
                        break ;
                    case java.sql.DatabaseMetaData.procedureColumnInOut:
                        if( datatype == java.sql.Types.OTHER ) {
                            stmt.setObject( sequence , arg , java.sql.Types.FLOAT , 0 );
                        } else {
                            stmt.registerOutParameter( sequence , datatype );
                        }
                        sequence ++ ;
                        break ;
                    case java.sql.DatabaseMetaData.procedureColumnResult:
                        if( 0 != paramCount ) { params += ", " ; }
                        params += ?;
                        paramCount ++ ;
                        break ;
                }
            }
        }
    }
```
protected static String connstr = "Connection";
protected static String clientstr = "org.okiproject.serviceimpl.mit.Client";
protected static int ROWS_IN_BLOCK = 1000;
private static final String EMPTYSTRING = "";
private static final String DEFAULTCONNECTION = EMPTYSTRING;
private String database = EMPTYSTRING;
private String getDatabase() {
    return database;
}
public String setDatabase( String database ) throws Exception {
    if ( !this.database.equals( database ) )
        resetCacheId();
    this.database = database;
    return database;
}
private static final String DEFAULTDATABASEURLPREFIX = "jdbc:oracle:oci8:@";
private String databaseURL = EMPTYSTRING;
public String getDatabaseURL() {
    return databaseURL;
}
private void setDatabaseURL( String databaseURL ) {
    if ( !this.databaseURL.equals( databaseURL ) )
        resetCacheId();
    this.databaseURL = databaseURL;
    return databaseURL;
}
private static final String DEFAULTDRIVERNAME = "oracle.jdbc.driver.OracleDriver";
private String driverName = DEFAULTDRIVERNAME;
public String getDriverName() {
    return driverName;
}
private void setDriverName( String driverName ) {
    this.driverName = driverName;
    return driverName;
}

Object[] onerow = new Object[] {
    Object[] columns;
    org.okiproject.serviceapi.sql.RowsMetaData rowsMetaData = new org.okiproject.serviceimpl.mit.sql.RowsMetaData( columnsMetaData ) ;
    org.okiproject.serviceapi.sql.Rows rows2 = new org.okiproject.serviceimpl.mit.sql.Rows( rowsMetaData, onerow ) ;
    org.okiproject.serviceapi.sql.Table table = org.okiproject.serviceimpl.mit.sql.Table( this, rows2 ) ;
    return table ;
}

protected static String connstr = "Connection";
protected static String clientstr = "org.okiproject.serviceimpl.mit.Client";
protected static int ROWS_IN_BLOCK = 1000;
private static final String EMPTYSTRING = "";
private static final String DEFAULTCONNECTION = EMPTYSTRING;
private String database = EMPTYSTRING;
private String getDatabase() {
    return database;
}
public String setDatabase( String database ) throws Exception {
    if ( !this.database.equals( database ) )
        resetCacheId();
    this.database = database;
    return database;
}
private static final String DEFAULTDATABASEURLPREFIX = "jdbc:oracle:oci8:@";
private String databaseURL = EMPTYSTRING;
public String getDatabaseURL() {
    return databaseURL;
}
private void setDatabaseURL( String databaseURL ) {
    if ( !this.databaseURL.equals( databaseURL ) )
        resetCacheId();
    this.databaseURL = databaseURL;
    return databaseURL;
}
private static final String DEFAULTDRIVERNAME = "oracle.jdbc.driver.OracleDriver";
private String driverName = DEFAULTDRIVERNAME;
public String getDriverName() {
    return driverName;
}
private void setDriverName( String driverName ) {
    this.driverName = driverName;
    return driverName;
}

private static final String DEFAULTDATABASEURLPREFIX = "jdbc:oracle:oci8:@";
private String databaseURL = EMPTYSTRING;
public String getDatabaseURL() {
    return databaseURL;
}
private void setDatabaseURL( String databaseURL ) {
    if ( !this.databaseURL.equals( databaseURL ) )
        resetCacheId();
    this.databaseURL = databaseURL;
    return databaseURL;
}
private static final String DEFAULTDRIVERNAME = "oracle.jdbc.driver.OracleDriver";
private String driverName = DEFAULTDRIVERNAME;
public String getDriverName() {
    return driverName;
}
private void setDriverName( String driverName ) {
    this.driverName = driverName;
    return driverName;
}
if ( null != username )
{
    this.username = username ;
}

private String password = EMPTYSTRING ;
private String getPassword() { return this.password ; }
private void setPassword( String password )
{
    if ( !this.password.equals( password ) )
    {
        resetCacheId();
        this.password = EMPTYSTRING ;
        if ( null != password )
        {
            this.password = password ;
        }
    }
}

private Boolean autoCommit = new Boolean( true ) ;
public Boolean getAutoCommit() { return this.autoCommit ; }
public void setAutoCommit( Boolean autoCommit )
{
    try
    {
        if ( !this.autoCommit.equals( autoCommit ) )
        {
            resetConnection();
            this.autoCommit = autoCommit ;
        }
    }
    catch( Exception ex )
    {
        ex.printStackTrace();
    }
}

private Connection()
{
    super() ;
}

public static org.okiproject.serviceimpl.mit.sql.Connection create
( String databaseURL , String driverName , String username , String password , String id )
throws Exception
{
    String exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDUSERNAME ;
    if ( null != username )
    {
        if ( null != password )
        {
            if ( null != databaseURL )
            {
new org.okiproject.serviceimpl.mit.sql.Connection();
                db.setDatabase( null ) ;
                db.setDatabaseURL( databaseURL ) ;
                db.setDriverName( driverName ) ;
                db.setUserName( username ) ;
                db.setPassword( password ) ;
                db.setId( id ) ;
                return db ;
            }
        else
        {                                                exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDDRIVERNAME ;
            }                                }
        else
        {                                        exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDDATABASEURL ;                                }
    else
    {                                exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDPASSWORD ;                        }
    }
    throw new org.okiproject.serviceapi.sql.ConnectionException( exStr ) ;
}

public static org.okiproject.serviceapi.sql.Connection create
( String database , String username , String password , String id )
throws Exception
{
    String exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDDATABASE ;
    if ( null != database )
    {
        if ( null != username )
        {
            if ( null != password )
            {
new org.okiproject.serviceimpl.mit.sql.Connection();
                db.setDatabase( database ) ;
                db.setDatabaseURL( null ) ;
                db.setDriverName( DEFAULTDRIVERNAME ) ;
                db.setUserName( username ) ;
                db.setPassword( password ) ;
                db.setId( id ) ;
                return db ;
            }
        else
        {                                                exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDUSERNAME ;
            }                                }
        else
        {                                        exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDPASSWORD ;                                }
    else
    {                                exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDDATABASEURL ;                        }
    else
    {                                exStr = org.okiproject.serviceapi.sql.ConnectionException.INVALIDDRIVERNAME ;                        }
    }
    throw new org.okiproject.serviceapi.sql.ConnectionException( exStr ) ;
}
org/okiproject/serviceimpl/mit/sql/Connection.java

```java
{ new org.okiproject.serviceapi.sql.ConnectionException( exStr ) ;
}

private String id = DEFAULTCONNECTION ;
public String getId() { return this.id ; }
private void setId( String id ) {
    this.id = DEFAULTCONNECTION ;
    if( null != id ) {
        this.id = id ;
    }
}

public synchronized boolean equals( org.okiproject.serviceapi.sql.Connection db ) {
    return
    { this == db 
    } ||
    { null != db 
        && ( db instanceof org.okiproject.serviceimpl.mit.sql.Connection )
    };
}

private Object cacheId = null ;
private synchronized Object getCacheId() {
    if( null == this.cacheId ) {
        objs[ 0 ] = getDatabase() ;
        objs[ 1 ] = getDatabaseURL() ;
        objs[ 2 ] = getUserName() ;
        objs[ 3 ] = getPassword() ;
        objs[ 4 ] = getId() ;
        objs[ 5 ] = connstr ;
        this.cacheId = objs ;
    }
    return this.cacheId ;
}
private void resetCacheId() { this.cacheId = null ; }

protected java.sql.Connection getConnection() throws Exception {
    try {
        java.sql.Connection conn = getConnectionFromCache() ;
        if( null == conn ) {
            String dbURL = DEFAULTDATABASEURLPREFIX + getDatabase() ;
            if( ! EMPTYSTRING.equals( getDatabaseURL() ) ) {
                dbURL = getDatabaseURL() ;
            }
            { getDriverName() , dbURL , getUserName() , getPassword() } ;
            conn = connect.getConnection() ;
            conn.setAutoCommit( conn.setAutoCommit().booleanValue() ) ;
            org.okiproject.serviceimpl.mit.sql.Connection db =
            Property( getCacheId() , conn ) ;
            return conn ;
        }
        catch( Exception ex ) {
            return null ;
        }
    }
    protected java.sql.Connection getConnectionFromCache() throws Exception {
        java.util.Enumeration enum =
        org.okiproject.serviceimpl.mit.util.RuntimeContext.getKeys() ;
        if( null != enum ) {
            while( enum.hasMoreElements() ) {
                Object[] test = ( Object[] ) enum.nextElement() ;
                if( getDatabase().equals( ( String ) test[ 0 ] )
                    && getDatabaseURL().equals( ( String ) test[ 1 ]
                    )
                    && getUserName().equals( ( String ) test[ 2 ]
                    )
                    && getPassword().equals( ( String ) test[ 3 ]
                    )
                    && getId().equals( ( String ) test[ 4 ]
                    )
                    && connstr.equals( ( String ) test[ 5 ]
                    )
                ) {
                    return ( java.sql.Connection )
                    org.okiproject.serviceimpl.mit.util.RuntimeContext.getProperty( test ) ;
                }
            }
        }
        return null ;
    }
```

protected void resetConnection()
    throws Exception
{
    java.sql.Connection conn
        = ( java.sql.Connection ) org.okiproject.serviceimpl.mit.util.RuntimeContext.getProperty
            ( getCacheId() ) ;
    if ( null != conn )
    {
        conn.close() ;
        org.okiproject.serviceimpl.mit.util.RuntimeContext.setProperty( getCacheId() ,
            null ) ;
    }
}

private String findDataTypeName( int datatype )
{
    for ( int i = 0 ; types.length != i ; i ++ )
    {
        if ( types[ i ] == datatype )
        {
            return typeNames[ i ] ;
        }
    }
    return null ;
}

private static int[] types =
{       java.sql.Types.ARRAY
        ,       java.sql.Types.BIGINT
        ,       java.sql.Types.BINARY
        ,       java.sql.Types.BIT
        ,       java.sql.Types.BLOB
        ,       java.sql.Types.CHAR
        ,       java.sql.Types.CLOB
        ,       java.sql.Types.DATE
        ,       java.sql.Types.DECIMAL
        ,       java.sql.Types.DISTINCT
        ,       java.sql.Types.DOUBLE
        ,       java.sql.Types.FLOAT
        ,       java.sql.Types.INTEGER
        ,       java.sql.Types.JAVA_OBJECT
        ,       java.sql.Types.LONGVARBINARY
        ,       java.sql.Types.LONGVARCHAR
        ,       java.sql.Types.NULL
        ,       java.sql.Types.NUMERIC
        ,       java.sql.Types.OTHER
        ,       java.sql.Types.REAL
        ,       java.sql.Types.REF
        ,       java.sql.Types.SMALLINT
        ,       java.sql.Types.STRUCT
        ,       java.sql.Types.TIME
        ,       java.sql.Types.TIMESTAMP
        ,       java.sql.Types.TINYINT
        ,       java.sql.Types.VARBINARY
        ,       java.sql.Types.VARCHAR
    } ;

private static String[] typeNames =
{       "java.sql.Types.ARRAY"
        ,       "java.sql.Types.BIGINT"
        ,       "java.sql.Types.BINARY"
        ,       "java.sql.Types.BIT"
        ,       "java.sql.Types.BLOB"
        ,       "java.sql.Types.CHAR"
        ,       "java.sql.Types.CLOB"
        ,       "java.sql.Types.DATE"
        ,       "java.sql.Types.DECIMAL"
        ,       "java.sql.Types.DISTINCT"
        ,       "java.sql.Types.DOUBLE"
        ,       "java.sql.Types.FLOAT"
        ,       "java.sql.Types.INTEGER"
        ,       "java.sql.Types.JAVA_OBJECT"
        ,       "java.sql.Types.LONGVARBINARY"
        ,       "java.sql.Types.LONGVARCHAR"
        ,       "java.sql.Types.NULL"
        ,       "java.sql.Types.NUMERIC"
        ,       "java.sql.Types.OTHER"
        ,       "java.sql.Types.REAL"
        ,       "java.sql.Types.REF"
        ,       "java.sql.Types.SMALLINT"
        ,       "java.sql.Types.STRUCT"
        ,       "java.sql.Types.TIME"
        ,       "java.sql.Types.TIMESTAMP"
        ,       "java.sql.Types.TINYINT"
        ,       "java.sql.Types.VARBINARY"
        ,       "java.sql.Types.VARCHAR"
package org.okiproject.serviceimpl.mit.sql;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute,
sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
<p>
The above copyright notice and this permission notice
shall be included in all copies or substantial portions of the
Software.
<p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

*/

public class Rows implements org.okiproject.serviceapi.sql.Rows {
  private Object[] rows = null;
  private int index = 0;
  private int offset = 0;

  private org.okiproject.serviceapi.sql.RowsMetaData rowsMetaData = null;
  public org.okiproject.serviceapi.sql.RowsMetaData getRowsMetaData()
  {
    return this.rowsMetaData;
  }

  public String[] getColumnNames()
    throws Exception
  {
    String[] columnNames = new String[ this.rowsMetaData.getColumnCount() ];
    for( int i = 0; columnNames.length != i; i++ )
    {
      org.okiproject.serviceapi.sql.ColumnMetaData colMetaData =
        this.rowsMetaData.getColumnsMetaData()[ i ];
      columnNames[ i ] = colMetaData.getColumnName();
    }
    return columnNames;
  }

  public int getColumn( String columnName )
    throws Exception
  {
    String[] columnNames = getColumnNames();
    for( int i = 0; columnNames.length != i; i++ )
    {
      if( columnName.equals( columnNames[ i ] ) )
      {
        return i;
      }
    }
    throw LOCALGETCOLUMNFAILED;
  }

  public boolean hasMoreRows()
    throws Exception
  {
    if( null != rows )
    {
      if( index < rows.length )
      {
        return true;
      }
      else
      {
        this.offset += index;
        index = 0;
        rows = null;
        if( null != table )
        {
          getRows();
          return( null != this.rows );
        }
      }
    }
    return false;
  }

  protected static int MAXNUMBEROFROWS = 1000;
  private static final Object[] NOARGS = new Object[ 0 ];
  private static final Class[] NOARGSTYPES = new Class[ 0 ];
  private static final String GETROWS = "getRows";
  private static final org.okiproject.serviceapi.sql.TableException LOCALGETROWSFAILED =
    new org.okiproject.serviceapi.sql.TableException( org.okiproject.serviceapi.sql.TableException.GETROWSFAILED ) ;
  private static final org.okiproject.serviceapi.sql.TableException LOCALGETCOLUMNFAILED =
    new org.okiproject.serviceapi.sql.TableException( org.okiproject.serviceapi.sql.TableException.GETCOLUMNFAILED ) ;

  private String catalog = null;
  private String schemaPattern = null;
  private String procedureNamePattern = null;
  private String columnNamePattern = null;
  private String tableNamePattern = null;
  private String[] tableTypes = null;

  protected org.okiproject.serviceimpl.mit.sql.Table table = null;
  private String sql = null;
  private Object[] objs = null;
  private int[] targetSqlTypes = null;
  private int[] scales = null;
}
public Object[] nextRow() throws Exception {
    if (null != rows) {
        return new Object[] { rows[index++] };  
    }
    throw LOCALGETROWSFAILED;  
}

public Object clone() {
    try {
        return new org.okiproject.serviceimpl.mit.sql.Rows(this.rowsMetaData, rows);  
    } catch (Exception ex) {
        ex.printStackTrace();  
    }
    return null;  
}

protected Rows {
    org.okiproject.serviceapi.sql.RowsMetaData rowsMetaData,
    Object[] rows  
} throws Exception {
    this.rowsMetaData = rowsMetaData;  
    this.rows = rows;  
}

protected Rows {
    org.okiproject.serviceimpl.mit.sql.Table table,
    String catalog,
    String schemaPattern,
    String tableNamePattern,
    String[] tableTypes  
} getRows();  

protected Rows {
    org.okiproject.serviceimpl.mit.sql.Table table,
    String sql,
    Object[] objs,
    int[] targetSqlTypes,
    int[] scales  
} getRows();

public void getRows() throws Exception {
    if (connection.hasClient()) {
        getRowsRemotely();  
    } else {
        getRowsLocally();  
    }  
}

private void getRowsRemotely() throws Exception {
    if (connection.hasClient()) {
        getRowsRemotely();  
    } else {
        getRowsLocally();  
    }  
}

protected Rows {
    org.okiproject.serviceimpl.mit.sql.Table table,
    String catalog,
    String schemaPattern,
    String procedureNamePattern  
} throws Exception {
    this.table = table;  
    this.catalog = catalog;  
    this.schemaPattern = schemaPattern;  
    this.procedureNamePattern = procedureNamePattern;  
    getRows();  
}
public void getRowsLocally()
    throws Exception
{
    java.sql.ResultSet rs = getResultSet();
    java.sql.ResultSetMetaData rsMeta = rs.getMetaData();
    this.rowsMetaData = new org.okiproject.serviceimpl.mit.sql.RowsMetaData(rsMeta);
    this.rows = createRows(rs);
    rs.close();
    statement.close();
}

protected java.sql.ResultSet getResultSet()
    throws Exception
{
    if (null != tableNamePattern) {
        return getTables();
    } else if (null != this.sql) {
        return getQuery();
    } else if (null != this.procedureName) {
        return getProcedureColumns();
    } else {
        return getProcedures();
    }
}

private java.sql.Connection getConnection()
    throws Exception
{
    org.okiproject.serviceapi.sql.Connection conn =
    { org.okiproject.serviceimpl.mit.sql.Table } this.table
    .getConnection();
    java.sql.Connection connection =
    { org.okiproject.serviceimpl.mit.sql.Connection } conn
    .getConnection();
    return connection;
}

private java.sql.ResultSet getTables()
    throws Exception
{
    java.sql.DatabaseMetaData meta = getConnection().getMetaData();
    return meta.getTables
    { this.catalog,
      this.schemaPattern,
      this.procedureName,
      this.tableNamePattern
    };
}

private java.sql.ResultSet getQuery()
    throws Exception
{
    java.sql.Connection connection =
    { org.okiproject.serviceimpl.mit.sql.Connection } conn
    .getConnection();
    java.sql.ResultSet rs = null;
    if (null == objs)
    {
        java.sql.Statement statement = connection.createStatement();
        rs = statement.executeQuery(sql);
    } else
    {
        java.sql.PreparedStatement statement = connection.prepareStatement(sql);
        for (int i = 0; objs.length != i; i++)
        {
            if (null != this.scales)
            {
                statement.setObject(i + 1, scales[i]);
            } else if (null != targetSqlTypes)
            {
                statement.setObject(i + 1, targetSqlTypes[i]);
            } else
            {
               (statement.setObject(i + 1, (null == objs[i]));
            }
        }
        rs = statement.executeQuery();
    }
    return rs;
}

private java.sql.ResultSet getProcedureColumns()
    throws Exception
{
    java.sql.DatabaseMetaData meta =
    { org.okiproject.serviceimpl.mit.sql.Connection } conn
    .getConnection();
    return meta.getProcedureColumns
    { this.catalog,
      this.schemaPattern,
      this.procedureName,
      this.tableNamePattern
    };
}
private java.sql.ResultSet getProcedures() throws Exception {
    java.sql.DatabaseMetaData meta = getConnection().getMetaData();
    return meta.getProcedures(new java.sql.Catalog(), new java.sql.SchemaPattern(), new java.sql.ProcedureNamePattern());
}

private Object[] createRows(java.sql.ResultSet rs) throws Exception {
    try {
        int rsType = rs.getType();
        if (rs.TYPE_FORWARD_ONLY == rsType) {
            if (rs.next()) {
                return createCells(rs);
            }
        } else if (rs.TYPE_SCROLL_INSENSITIVE == rsType) {
            rs.setFetchDirection(rs.FETCH_FORWARD);
            if (rs.first()) {
                return createCells(rs);
            }
        } else if (rs.TYPE_SCROLL_SENSITIVE == rsType) {
            rs.setFetchDirection(rs.FETCH_FORWARD);
            boolean isRowAvailable = rs.first();
            if (rs.first()) {
                return createCells(rs);
            }
        }
    } catch (Exception ex) {
        String message = LOCALGETROWSFAILED.getMessage();
        message += ex.getMessage();
        throw new org.okiproject.serviceapi.sql.TableException(message);
    }
    return null;
}

private Object createCell(Object obj, int row, int col) throws Exception {
    Object cell = obj;
    if (obj instanceof java.sql.Blob) {
        cell = makeBlob((java.sql.Blob) obj, row, col);
    } else if (obj instanceof java.sql.Clob) {
        cell = makeClob((java.sql.Clob) obj, row, col);
    }
    return cell;
}

private Object createCells(java.sql.ResultSet rs) throws Exception {
    Object[] rows = new Object[MAXNUMBEROFROWS];
    int cursor = 0;
    int rowsAdded = 0;
    boolean isRowAvailable = true;
    while (isRowAvailable) {
        if (cursor >= this.offset) {
            Object[] columns = new Object[this.rowsMetaData.getColumnCount()];
            for (int i = 0; columns.length != i; i++) {
                columns[i] = createCell(rs.getObject(i + 1), rowsAdded, i);
            }
            rows[rowsAdded] = columns;
            rowsAdded++;
        }
    }
    if (MAXNUMBEROFROWS == rowsAdded) {
        break;
    }
    cursor++;
    isRowAvailable = rs.next();
    if (0 == rowsAdded) {
        return null;
    }
    if (MAXNUMBEROFROWS > rowsAdded) {
        Object[] newRows = new Object[rowsAdded];
        for (int j = 0; rowsAdded != j; j++) {
            newRows[j] = rows[j];
        }
        rows = newRows;
    }
    return rows;
}

private Object makeBlob(java.sql.Blob blob, int row, int col) throws Exception {
}
private org.okiproject.serviceapi.sql.Clob makeClob
    ( java.sql.Clob clob,
    int row,
    int col )
    throws Exception
{
    return new org.okiproject.serviceimpl.mit.sql.Clob
        ( ( org.okiproject.serviceimpl.mit.sql.Rows )
            this.clone()
        , clob.length(), row, col )
    ;
}
package org.okiproject.serviceimpl.mit.sql;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:</p><p>The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.</p><p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.</p>*/

public class RowsMetaData implements org.okiproject.serviceapi.sql.RowsMetaData {
    protected int columnCount = 0;
    public int getColumnCount() { return this.columnCount; }

    protected org.okiproject.serviceapi.sql.ColumnMetaData[] columnsMetaData = null;
    public org.okiproject.serviceapi.sql.ColumnMetaData[] getColumnsMetaData() { return this.columnsMetaData; }

    protected RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    protected RowsMetaData(org.okiproject.serviceapi.sql.ColumnMetaData[] columnsMetaData) {
        if (null != columnsMetaData) {
            this.columnsMetaData = columnsMetaData;
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }

    public RowsMetaData(java.sql.ResultSetMetaData rsMeta) throws Exception {
        this.columnCount = rsMeta.getColumnCount();
        if (0 != this.columnCount) {
            this.columnsMetaData = new org.okiproject.serviceapi.sql.ColumnMetaData[
                this.columnCount];
            for (int i = 0; i < this.columnsMetaData.length; i++) {
                this.columnsMetaData[i] = new org.okiproject.serviceimpl.mit.sql.ColumnMetaData(i + 1, rsMeta);
            }
        }
    }

    public RowsMetaData() {
    }
package org.okiproject.serviceimpl.mit.sql;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p>The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.</p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.<p>*/

public class SqlFactory
extends org.okiproject.serviceapi.sql.SqlFactory
{
    public org.okiproject.serviceapi.sql.Connection newConnection
    ( String databaseURL                , String driverName                , String username                , String password                , String id                )
throws Exception
    
    
    return org.okiproject.serviceimpl.mit.sql.Connection.create
    ( databaseURL
      , driverName
      , username
      , password
      , id
    ) ;

    public org.okiproject.serviceapi.sql.Connection newConnection
    ( String database                , String username                , String password                , String id                )
throws Exception
    
    
    return org.okiproject.serviceimpl.mit.sql.Connection.create
    ( database
      , username
      , password
      , id
    ) ;

    public org.okiproject.serviceapi.sql.Connection newConnection
    ( java.util.Properties properties                , String id                )
throws Exception
    
    
    org.okiproject.serviceapi.sql.Connection connection = null ;
    if( null != properties )
    {
        String username = ( String ) properties.get( "Username" ) ;
        String password = ( String ) properties.get( "Password" ) ;
        if( null != properties.get( "databaseURL" ) )
        {
            String databaseURL = ( String ) properties.get( "databaseURL" ) ;
            String driverName = ( String ) properties.get( "driver Name" ) ;
            connection =
            org.okiproject.serviceimpl.mit.sql.Connection.create
            ( databaseURL
              , driverName
              , username
              , password
              , id
            ) ;
        }
        else
        {
            String database = ( String ) properties.get( "Database " ) ;
            connection =
            org.okiproject.serviceimpl.mit.sql.Connection.create
            ( database
              , username
              , password
              , id
            ) ;
        }
    }
    return connection ;
}
package org.okiproject.serviceimpl.mit.sql;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>   person obtaining a copy of this software and associated<p>   documentation files (the "Software"), to deal in the<p>   Software without restriction, including without limitation<p>   the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>   Software, and to permit persons to whom the Software is<p>   furnished to do so, subject to the following conditions:<p>   The above copyright notice and this permission notice<p>   shall be included in all copies or substantial portions of<p>   the Software.  
<p>   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p>   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>   WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>   PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>   DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>   CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>   IN THE SOFTWARE.</p>*/

public class Table implements org.okiproject.serviceapi.sql.Table
{
    private static final org.okiproject.serviceapi.sql.TableException GETTABLEFAIL
        = new org.okiproject.serviceapi.sql.TableException("GETTABLEFAIL" );

    private org.okiproject.serviceapi.sql.Connection connection = null;
    public org.okiproject.serviceapi.sql.Connection getConnection()
    {
        return this.connection ;
    }

    private org.okiproject.serviceapi.sql.Rows rows = null;
    public org.okiproject.serviceapi.sql.Rows getRows()
    {
        if (null != this.rows )
        {
            return new org.okiproject.serviceapi.sql.Rows()
            {
                { org.okiproject.serviceapi.sql.Rows rows = this.rows.clone();
                }
            };
        }
        return null;
    }

    public static org.okiproject.serviceapi.sql.Table getTables
    { org.okiproject.serviceapi.sql.Connection connection
        throws Exception
        {
            String[] tableTypes[ ] = { "TABLE" ;
                return getTables
                {
                    connection
                    , null
                    , null
                    , null
                    , tableTypes
                } ;
        }

    public static org.okiproject.serviceapi.sql.Table getTables
    { org.okiproject.serviceapi.sql.Connection connection
        String catalog
        , String schemaPattern
        , String tableNamePattern
        , String[] tableTypes
        throws Exception
        {
            return new org.okiproject.serviceimpl.mit.sql.Table
            {
                connection
                , catalog
                , schemaPattern
                , tableNamePattern
                , tableTypes
            } ;
        }

    public static org.okiproject.serviceapi.sql.Table executeQuery
    { org.okiproject.serviceapi.sql.Connection connection
        String sql
        , Object[] objs
        throws Exception
        {
            return new org.okiproject.serviceimpl.mit.sql.Table
            {
                connection
                , sql
                , objs
                , ( int[] ) null
                , ( int[] ) null
            } ;
        }

    public static org.okiproject.serviceapi.sql.Table executeQuery
    { org.okiproject.serviceapi.sql.Connection connection
        String sql
        , Object[] objs
        throws Exception
        {
            return new org.okiproject.serviceimpl.mit.sql.Table
            {
                connection
                , sql
                , objs
                , ( int[] ) null
                , ( int[] ) null
            } ;
        }
public static org.okiproject.serviceapi.sql.Table executeQuery
    (org.okiproject.serviceapi.sql.Connection connection,
     String sql,
     Object[] objs,
     int[] targetSqlTypes,
     int[] scales)
    throws Exception
    {
        org.okiproject.serviceapi.sql.Table table =
            new org.okiproject.serviceimpl.mit.sql.Table
                (connection,
                 sql,
                 objs,
                 targetSqlTypes,
                 scales);
        return table;
    }

protected Table
    (org.okiproject.serviceapi.sql.Connection connection,
     org.okiproject.serviceapi.sql.Rows rows)
    throws Exception
    {
        this.connection = connection;
        this.rows = rows;
    }

private Table
    (org.okiproject.serviceapi.sql.Connection connection,
     String catalog,
     String schemaNamePattern,
     String tableNamePattern,
     String[] tableTypes)
    throws Exception
    {
        this.connection = connection;
        this.rows = new org.okiproject.serviceimpl.mit.sql.Rows
            (this,
             catalog,
             schemaNamePattern,
             tableNamePattern,
             tableTypes);
    }

private Table
    (org.okiproject.serviceapi.sql.Connection connection,
     String sql,
     Object[] objs,
     int[] targetSqlTypes,
     int[] scales)
    throws Exception
    {
        this.connection = connection;
        this.rows = new org.okiproject.serviceimpl.mit.sql.Rows
            (this,
             sql,
             objs,
             targetSqlTypes,
             scales);
    }

public static org.okiproject.serviceapi.sql.Table getProcedures
    (org.okiproject.serviceapi.sql.Connection connection,
     String catalog,
     String schemaPattern,
     String procedureNamePattern)
    throws Exception
    {
        // code implementation
    }

private Table
    (org.okiproject.serviceapi.sql.Connection connection,
     String sql,
     Object[] objs,
     int[] targetSqlTypes,
     procedureNamePattern)
    throws Exception
    {
        // code implementation
    }

public static org.okiproject.serviceimpl.mit.sql/Table.java
    ( connection,
      catalog,
      schemaPattern,
      procedureNamePattern
    )
    
    return table;
}
public class Table {  
    private Table(  
        org.okiproject.serviceapi.sql.Connection connection,  
        String catalog,  
        String schemaPattern,  
        String procedureName,  
        String columnNamePattern  
    ) throws Exception {  
        this.connection = connection;  
        this.rows = new org.okiproject.serviceimpl.mit.sql.Rows  
            (this, catalog, schemaPattern, procedureName, columnNamePattern);  
    }  
}

private Table(  
    org.okiproject.serviceapi.sql.Connection connection,  
    String catalog,  
    String schemaPattern,  
    String procedureName,  
    String columnNamePattern  
) throws Exception {  
    this.connection = connection;  
    this.rows = new org.okiproject.serviceimpl.mit.sql.Rows  
        (this, catalog, schemaPattern, procedureName, columnNamePattern);  
}

}
package org.okiproject.serviceimpl.mit.util;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:<p>The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.</p></p>*/
public class RuntimeContext {
    private static java.util.Hashtable contexts = new java.util.Hashtable();

    public static boolean contains( Object key )
    {
        try
        {
            if ( null != key )
            {
                Runtime runtime = Runtime.getRuntime();
                if ( contexts.containsKey( runtime ) )
                {
                    java.util.Hashtable context = ( java.util.Hashtable ) contexts.get( runtime );
                    return context.containsKey( key );
                }
            }
            catch( Exception ex )
            {
                return false;
            }
        }
        return false;
    }

    public static void setProperty( Object key, Object value )
    {
        try
        {
            if ( null != key )
            {
                Runtime runtime = Runtime.getRuntime();
                java.util.Hashtable context = ( java.util.Hashtable ) contexts.get( runtime );
                if ( null != value )
                {
                    context.put( key, value );
                }
                else
                {
                    context.remove( key );
                }
            }
            catch( Exception ex )
            {
                ex.printStackTrace();
            }
        }
        }
    }

    public static Object getProperty( Object key )
    {
        try
        {
            if ( null != key )
            {
                Runtime runtime = Runtime.getRuntime();
                if ( contexts.containsKey( runtime ) )
                {
                    java.util.Hashtable context = ( java.util.Hashtable ) contexts.get( runtime );
                    return context.get( key );
                }
            }
            catch( Exception ex )
            {
                ex.printStackTrace();
            }
            return null;
        }
    }

    public static java.util.Enumeration getKeys()
    {
        try
        {
            Runtime runtime = Runtime.getRuntime();
            if ( contexts.containsKey( runtime ) )
            {
                java.util.Hashtable context = ( java.util.Hashtable ) contexts.get( runtime );
                return context.keys();
            }
            catch( Exception ex )
            {
                ex.printStackTrace();
            }
            return null;
        }
    }
}

if (! contexts.containsKey( runtime ) )
{
    contexts.put( runtime, new java.util.Hashtable() );
}
return null;

public static Object removeProperty(String key) {
    try {
        if (null != key) {
            Runtime runtime = Runtime.getRuntime();
            if (!contexts.containsKey(runtime)) {
                java.util.Hashtable context = (java.util.Hashtable) contexts.get(runtime);
                if (context.containsKey(key)) {
                    Object value = context.get(key);
                    if (value instanceof java.sql.Connection) {
                        ( (java.sql.Connection) value ).close();
                    } else if (value instanceof java.sql.ResultSet) {
                        ( (java.sql.ResultSet) value ).close();
                    }
                    Object obj = context.remove(key);
                    System.gc();
                    return obj;
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
            return null;
        }
    }
    return null;
}
package org.okiproject.serviceimpl.mit.util;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * 
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public class ThreadContext extends Thread {

    private static java.util.Hashtable contexts = new java.util.Hashtable();

    private static void gc() {
        java.util Enumeration keys = contexts.keys();
        while (keys.hasMoreElements()) {
            Thread thread = (Thread) keys.nextElement();
            contexts.remove(thread);
        }
        System.gc();
    }

    public static void setProperty(Object key, Object value) {
        try {
            gc();
            if (null != key) {
                Thread thread = Thread.currentThread();
                if (! contexts.containsKey(thread)) {
                    contexts.put(thread, new java.util.Hashtable());
                }
                contexts.get(thread);
                if (null != value) {
                    context.put(key, value);
                } else {
                    context.remove(key);
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    public static Object getProperty(Object key) {
        try {
            gc();
            if (null != key) {
                Thread thread = Thread.currentThread();
                if (contexts.containsKey(thread)) {
                    java.util.Hashtable context = (java.util.Hashtable) contexts.get(thread);
                    return context.get(key);
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return null;
    }

    public static java.util Enumeration getPropertyName() {
        try {
            gc();
            Thread thread = Thread.currentThread();
            if (! contexts.containsKey(thread)) {
                java.util.Hashtable context = (java.util.Hashtable) contexts.get(thread);
            return context.keys();
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return null;
    }

    public static Object removeProperty(String name) {
        try {
            gc();
            Thread thread = Thread.currentThread();
            if (! contexts.containsKey(thread)) {
                java.util.Hashtable context = (java.util.Hashtable) contexts.get(thread);
            return context.remove(name);
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return null;
    }
}
gc();
if(null != name)
{
    Thread thread = Thread.currentThread();
    if(!contexts.containsKey(thread))
    {
        java.util.Hashtable context = (java.util.Hashtable)contexts.get(thread);
        return(context.remove(name));
    }
}
}
catch(Exception ex)
{
    ex.printStackTrace();
}
return null;

public static void purge()
{
    try
    {
        java.util.Enumeration threads = contexts.keys();
        while(threads.hasMoreElements())
        {
            java.util.Hashtable context = (java.util.Hashtable)threads.nextElement();
            java.util.Enumeration keys = context.keys();
            while(keys.hasMoreElements())
            {
                Object key = keys.nextElement();
                context.remove(key);
            }
            contexts.remove(context);
            contexts = new java.util.Hashtable();
        }
    }
    catch(Exception ex)
    {
        ex.printStackTrace();
    }
    System.gc();
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 */

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

/**
 @Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/ClassadminException.java,v
 @Version: 1.3
 @Date: 2002/02/08 14:25:41
 */

public class ClassadminException
extends java.lang.Exception {
    public static final String UNIMPLEMENTED = "Unimplemented Classadmin method ";
    public static final String DEFAULTCLASSADMINFACTORYDIDNOTLOAD = "Default Classadmin factory did not load ";
    public static final String CLASSADMINFACTORYPROPERTIESDIDNOTLOAD = "Classadmin factory properties did not load ";

    /**
     * Constructor ClassadminException
     *
     * @param message
     *
     */
    public ClassadminException(String message) {
        super(message);
    }
}
package org.okiproject.okifuncapi.classadmin;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>   person obtaining a copy of this software and associated<p>   documentation files (the "Software"), to deal in the<p>   Software without restriction, including without limitation<p>   the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>   Software, and to permit persons to whom the Software is<p>   furnished to do so, subject to the following conditions:<p>   The above copyright notice and this permission notice<p>   shall be included in all copies or substantial portions of the<p>   Software.<p>   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p>   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>   WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>   PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>   DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>   CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>   IN THE SOFTWARE.<p>*/

public class ClassadminFactory
    implements java.io.Serializable
{
    private static final org.okiproject.okifuncapi.classadmin.ClassadminException UNIMPLEMENTED
        = new org.okiproject.okifuncapi.classadmin.ClassadminException()
            ( org.okiproject.okifuncapi.classadmin.ClassadminException.UNIMPLEMENTED);

    public static org.okiproject.okifuncapi.classadmin.ClassadminFactory getDefaultClassadminFactory()
        throws Exception
    {
        java.util.Map properties = getPropertiesFromPRB();
        return getClassadminFactory( properties );
    }

    public static org.okiproject.okifuncapi.classadmin.ClassadminFactory getClassadminFactory
        ( java.util.Map props )
        throws Exception
    {
        java.util.Map properties = props
        if ( null == properties )
        {
            properties = getPropertiesFromPRB();
        }
        if ( null != properties )
        {
            String factoryClassName = (String) properties.get( "ClassadminFactory" );
            if ( null != factoryClassName )
            {
                Class factoryClass
                    = Class.forName( factoryClassName );
                org.okiproject.okifuncapi.classadmin.ClassadminFactory
                    factoy
                        = ( org.okiproject.okifuncapi.classadmin.ClassadminFactory )
                            factoryClass.newInstance();
                factoryClass.setProperties( properties ) ;
                return factory ;
            }
        }

        throw new org.okiproject.okifuncapi.classadmin.ClassadminException()
            ( org.okiproject.okifuncapi.classadmin.ClassadminException.DEFAULTCLASSADMINFACTORYDIDNOTLOAD );
    }

    private java.util.Map properties = null;
    protected java.util.Map getProperties()
    {
        return this.properties ;
    }

    protected void setProperties( java.util.Map properties )
    {
        this.properties = properties ;
    }

    protected static java.util.Properties getPropertiesFromPRB()
        throws Exception
    {
        java.util.PropertyResourceBundle prb
            = ( java.util.PropertyResourceBundle )
                java.util.ResourceBundle.getBundle( "ClassadminFactory" );
        if ( null != prb )
        {
            java.util.Properties properties
                = new java.util.Properties();
            java.util.Enumeration keys = prb.getKeys();
            while ( keys.hasMoreElements() )
            {
                String key = ( String ) keys.nextElement();
                properties.setProperty( key , ( String ) prb.handleGetObject( key ) );
            }
        }
        return properties ;
    }

    throw new org.okiproject.okifuncapi.classadmin.ClassadminException()
        ( org.okiproject.okifuncapi.classadmin.ClassadminException.CLASSADMINFACTORYPROPERTIESDIDNOTLOAD );
}

/**
   Returns a new instance of a Course that has been created in the database
   Create the authorization context using the GUID created
   */
/** 
 * Create a Course
 */
public org.okiproject.okifuncapi.classadmin.Course createCourse( /*details*/ )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Destroy a Course
 */
public void destroyCourse( org.okiproject.okifuncapi.classadmin.Course course
)
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Returns Courses
 */
public org.okiproject.okifuncapi.classadmin.Courses getCourses( /*details*/ )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Destroy Courses
 */
public destroyCourses( org.okiproject.okifuncapi.classadmin.Courses courses )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Returns a new instance of a Person that has been created in the database
 */
public org.okiproject.okifuncapi.classadmin.Person createPerson( /*details*/ )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Returns a new instance of a Student that has been created in the database
 */
public org.okiproject.okifuncapi.classadmin.Student createStudent( /*details*/ )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Returns a new instance of a Student
 */
public org.okiproject.okifuncapi.classadmin.Student newStudent( /*details*/ )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Destroy a Student
 */
public void destroyStudent( org.okiproject.okifuncapi.classadmin.Student student
)
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Returns Students
 */
public org.okiproject.okifuncapi.classadmin.Students getStudents( /*details*/ )
throws Throwable
{
    throw UNIMPLEMENTED;
}
/** 
 * Destroy Students
 */
public void destroyStudents( org.okiproject.okifuncapi.classadmin.Students students
)
throws Throwable
{
```java
/**
 * Returns a new instance of a Term that has been created in the database
 */
public org.okiproject.okifuncapi.classadmin.Term createTerm(/*details*/)
 throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns a new instance of a Term
*/
public org.okiproject.okifuncapi.classadmin.Term newTerm(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Destroy a Term
*/
public void destroyTerm( org.okiproject.okifuncapi.classadmin.Term term )
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns Terms
*/
public org.okiproject.okifuncapi.classadmin.Terms getTerms(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Destroy Terms
*/
public void destroyTerms( org.okiproject.okifuncapi.classadmin.Terms terms )
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns a new instance of a Grade that has been created in the database
*/
public org.okiproject.okifuncapi.classadmin.Grade createGrade(/*details*/)
 throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns a new instance of a Grade
*/
public org.okiproject.okifuncapi.classadmin.Grade newGrade(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Destroy a Grade
*/
public void destroyGrade( org.okiproject.okifuncapi.classadmin.Grade grade )
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns Grades
*/
public org.okiproject.okifuncapi.classadmin.Grades getGrades(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Destroy Grades
*/
public void destroyGrades( org.okiproject.okifuncapi.classadmin.Grades grades )
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns a new instance of a Gradebook that has been created in the database
*/
public org.okiproject.okifuncapi.classadmin.Gradebook createGradebook(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns a new instance of a Gradebook
*/
public org.okiproject.okifuncapi.classadmin.Gradebook newGradebook(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Returns Gradebooks
*/
public org.okiproject.okifuncapi.classadmin.Gradebooks getGradebooks(/*details*/)
throws Throwable
{
    throw UNIMPLEMENTED;
}

/**
Destroy Gradebooks
*/
public void destroyGradebooks( org.okiproject.okifuncapi.classadmin.Gradebooks
```
throw UNIMPLEMENTED;
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED \"AS IS\", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

@version 1.2 $Revision: 1.2 $ Date: 2002/02/04 23:25:29 $
*/

public interface Course
{
    org.okiproject.okifuncapi.content.Asset getContentHome ();
    void setContentHome (org.okiproject.okifuncapi.content.LearningAsset contentHome);
    org.okiproject.okifuncapi.classadmin.Students getStudents ();
    void setStudents (org.okiproject.okifuncapi.classadmin.Students students);
    org.okiproject.okifuncapi.classadmin.Students getRegisteredStudents ();
    void setRegisteredStudents (org.okiproject.okifuncapi.classadmin.Students students);
    org.okiproject.okifuncapi.classadmin.Term getTerm ();
    void setTerm (org.okiproject.okifuncapi.classadmin.Term term);
    org.okiproject.okifuncapi.classadmin.Schedule getSchedule ();
    void setSchedule (org.okiproject.okifuncapi.classadmin.Schedule schedule);
    org.okiproject.okifuncapi.classadmin.Courses getPrerequisits ();
    void setPrerequisites (org.okiproject.okifuncapi.classadmin.Courses courses);
    org.okiproject.okifuncapi.classadmin.Grades getGrades ();
    void setGrades (org.okiproject.okifuncapi.classadmin.Grades grades);
    //[chs 02/04/02] what type is Major?
    Major getMajor();
    void setMajor (Major major);
    CanonicalCourse getCanonicalCourse ();
    void setCanonicalCourse (CanonicalCourse canonicalCourse);
    /* or should this be parents & children with a get/set coursetype which dete-
mines ***/
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Courses
extends java.io.Serializable {

/**
 * Method add
 *
 * @param course
 *
 */
    void add(org.okiproject.okifuncapi.classadmin.Course course);

/**
 * Method remove
 *
 * @param course
 *
 */
    void remove(org.okiproject.okifuncapi.classadmin.Course course);

/**
 * Method iterator
 *
 * @return org.okiproject.okifuncapi.classadmin.CoursesIterator
 *
 */
    org.okiproject.okifuncapi.classadmin.CoursesIterator iterator();}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface CoursesIterator
    extends java.io.Serializable {

    /**
     * Method hasMore
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     * @return org.okiproject.okifuncapi.classadmin.Course
     */
    org.okiproject.okifuncapi.classadmin.Course next();
    }

package org.okiproject.okifuncapi.course;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Grade extends java.io.Serializable
package org.okiproject.okifuncapi.course;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology</p>
<p>Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:</p>
<p>The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.</p>
<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.</p>
*/

public interface Gradebook
{
    implements java.io.Serializable
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Gradebooks extends java.io.Serializable {

/**
 * Method add
 */

void add(org.okiproject.okifuncapi.classadmin.Gradebook gradebook);

/**
 * Method remove
 */

void remove(org.okiproject.okifuncapi.classadmin.Gradebook gradebook);

/**
 * Method iterator
 */

@org.okiproject.okifuncapi.classadmin.GradebooksIterator iterator();
}
package org.okiproject.okifuncapi.classadmin;

/**
   * Copyright (c) 2001 Massachusetts Institute of Technology
   * Permission is hereby granted, free of charge, to any
   * person obtaining a copy of this software and associated
   * documentation files (the "Software"), to deal in the
   * Software without restriction, including without limitation
   * the rights to use, copy, modify, merge, publish,
   * distribute, sublicense, and/or sell copies of the
   * Software, and to permit persons to whom the Software is
   * furnished to do so, subject to the following conditions:
   * The above copyright notice and this permission notice
   * shall be included in all copies or substantial portions of
   * the Software.
   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
   * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
   * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
   * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
   * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
   * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
   * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
   * IN THE SOFTWARE.
   */

Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/GradebooksIterator.java,v $ @version $ @Revision: 1.2 $ @date: 2002/02/08 14:25:41 $ */

public interface GradebooksIterator
extends java.io.Serializable {

/**
   * Method hasMore
   *
   * @return Boolean
   */
   Boolean hasMore();

/**
   * Method next
   *
   * @return org.okiproject.okifuncapi.classadmin.Gradebook
   */
   org.okiproject.okifuncapi.classadmin.Gradebook next();
}
package org.okiproject.okifuncapi.course;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

$$Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/Grades.java,v $$
@version@ $Revision: 1.3 $$
@date: 2002/02/08 14:25:41 $*/

public interface Grades
extends java.io.Serializable {

/**
 * Method add
 *
 * @param grade
 */
 void add(org.okiproject.okifuncapi.classadmin.Grade grade);

/**
 * Method remove
 *
 * @param grade
 */
 void remove(org.okiproject.okifuncapi.classadmin.Grade grade);

/**
 * Method iterator
 *
 * @return org.okiproject.okifuncapi.classadmin.GradesIteratory
 */
 org.okiproject.okifuncapi.classadmin.GradesIteratory iterator();}
package org.okiproject.okifuncapi.course;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

Source: /cvs/oki/org.okiproject.okifuncapi/classadmin/GradesIterator.java,v $
@version $Revision: 1.2 $ $Date: 2002/02/08 14:25:41 $
*/
public interface GradesIterator
  extends java.io.Serializable {

  /**
   * Method hasMore
   *
   * @return Boolean
   */
  Boolean hasMore();

  /**
   * Method next
   *
   * @return org.okiproject.okifuncapi.classadmin.Grade
   */
  org.okiproject.okifuncapi.classadmin.Grade next();
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/People.java,v $ @version@ $Revision: 1.3 $ $Date: 2002/02/08 14:25:41 $
public interface People
extends java.io.Serializable {

/**
 * Method add
 * 
 * @param person
 */
void add(org.okiproject.okifuncapi.classadmin.Person person);

/**
 * Method remove
 * 
 * @param person
 */
void remove(org.okiproject.okifuncapi.classadmin.Person person);

/**
 * Method iterator
 * 
 * @return org.okiproject.okifuncapi.classadmin.PeopleIterator
 */
org.okiproject.okifuncapi.classadmin.PeopleIterator iterator();}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * 
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * 
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface PeopleIterator
    extends java.io.Serializable {
    Boolean hasMore();
    org.okiproject.okifuncapi.classadmin.Person next();
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

public interface Person
    extends java.io.Serializable
{
    String getAlias();
    void setAlias( String alias );
    Roles getRoles();
    void setRoles( Roles roles );
    Preferences getPreferences();
    void setPreferences( Preferences preferences );
    Personinfo getPersonInfo();
    void setPersoninfo(Personinfo personinfo);
    void refresh();
    void update();
    void delete();
    Timestamp getModifyDate();
    org.okiproject.okifuncapi.classadmin.Person getModifyUser();
    Timestamp getCreateDate();
    Object getId();
    Work_queue getMessages();
    Message_queue getWork();
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Student
    extends org.okiproject.okifuncapi.classadmin.Person {

    /**
     * Method getGrades
     *
     * @return org.okiproject.okifuncapi.classadmin.Grades
     */
    org.okiproject.okifuncapi.classadmin.Grades getGrades();

    /**
     * Method getSchedule
     *
     * @return org.okiproject.okifuncapi.utility.Schedule
     */
    org.okiproject.okifuncapi.utility.Schedule getSchedule();

    /**
     * Method getContent
     *
     * @return org.okiproject.okifuncapi.content.LearningAsset
     */
    org.okiproject.okifuncapi.content.LearningAsset getContent();
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * 
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * 
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 *
**/ $Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/Students.java,v $ @version@ @Revision: 1.4 $ @date: 2002/02/08 14:25:41 $ */

public interface Students extends java.io.Serializable {

/**
 * Method add
 * 
 * @param student
 */

void add(org.okiproject.okifuncapi.classadmin.Student student);

/**
 * Method remove
 * 
 * @param student
 */

void remove(org.okiproject.okifuncapi.classadmin.Student student);

/**
 * Method iterator
 * 
 * @return org.okiproject.okifuncapi.classadmin.StudentsIterator
 */

org.okiproject.okifuncapi.classadmin.StudentsIterator iterator();
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * 
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * 
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 * 
 **/ $Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/StudentsIterator.java,v $ $Revision: 1.2 $ $Date: 2002/02/08 14:25:41 $ */

public interface StudentsIterator extends java.io.Serializable {

    /**
     * Method hasMore
     *
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     *
     * @return org.okiproject.okifuncapi.classadmin.Student
     */
    org.okiproject.okifuncapi.classadmin.Student next();
}
package org.okiproject.okifuncapi.classadmin;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * 
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * 
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Term extends java.io.Serializable {

    /**
     * Method getDates
     *
     * @return Dates
     */
    Dates getDates();

    /**
     * Method setDates
     *
     * @param dates
     */
    void setDates(Dates dates);
}
package org.okiproject.okifuncapi.classadmin;

/**
*p> Copyright (c) 2001 Massachusetts Institute of Technology
<p>
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
<p>
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
<p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
<p>
*/
public interface Terms
extends java.io.Serializable {

/**
 * Method add
 * @param term
 */
void add(org.okiproject.okifuncapi.classadmin.Term term);

/**
 * Method remove
 * @param term
 */
void remove(org.okiproject.okifuncapi.classadmin.Term term);

/**
 * Method iterator
 * @return org.okiproject.okifuncapi.classadmin.TermsIterator
 */
org.okiproject.okifuncapi.classadmin.TermsIterator iterator();}
//**
<p>Copyright (c) 2001 Massachusetts Institute of Technology
<p>Permission is hereby granted, free of charge, to any
person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
<p>The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, 
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS 
IN THE SOFTWARE.
<p>$Source: /cvs/oki/org/okiproject/okifuncapi/classadmin/TermsIterator.java,v $
$Revision: 1.2 $
$Date: 2002/02/08 14:25:41 $
**/
public interface TermsIterator
extends java.io.Serializable {

/**
 * Method hasMore
 *
 * @return Boolean
 */
Boolean hasMore();

/**
 * Method next
 *
 * @return org.okiproject.okifuncapi.classadmin.Term
 */
org.okiproject.okifuncapi.classadmin.Term next();}
package org.okiproject.okifuncapi.content;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>   person obtaining a copy of this software and associated<p>   documentation files (the "Software"), to deal in the<p>   Software without restriction, including without limitation<p>   the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>   Software, and to permit persons to whom the Software is<p>   furnished to do so, subject to the following conditions:<p>   The above copyright notice and this permission notice<p>   shall be included in all copies or substantial portions of<p>   the Software.</p>   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p>   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>   WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>   PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>   DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>   CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>   IN THE SOFTWARE.</p>   */

public interface Asset
    extends java.io.Serializable
{
    Timestamp getEffectiveDate();
    Timestamp getExpirationDate();
    Timestamp getModifyDate();
    Timestamp getCreateDate();
    Object getId();
    String getName();
    String getDescription();
    Status getStatus();
    void update(field list...)
    List Properties?
    Refresh();
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
<p>
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
</p>
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
</p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

*/
public class ContentException extends java.lang.Exception {
    public static final String UNIMPLEMENTED = "Unimplemented Content method ";
    public static final String DEFAULTCONTENTFACTORYDIDNOTLOAD = "Default Content factory did not load ";
    public static final String CONTENTFACTORYPROPERTIESDIDNOTLOAD = "Content factory properties did not load ";

    /**
     * Constructor ContentException
     * @param message
     */
    public ContentException(String message) {
        super(message);
    }
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:
**
 * The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.
**
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
**
 * Source: /cvs/oki/org.okiproject.okifuncapi/content/ContentFactory.java,v $
 * @version $Revision: 1.3 $ $Date: 2002/02/08 14:25:41 $ */

public class ContentFactory
    implements java.io.Serializable {
    private static final org.okiproject.okifuncapi.content.ContentException UNIMPLEMENTED =
        new org.okiproject.okifuncapi.content.ContentException(
            org.okiproject.okifuncapi.content.ContentException.UNIMPLEMENTED);

    /**
     * @return org.okiproject.okifuncapi.content.ContentFactory
     * @throws Exception
     */
    public static org.okiproject.okifuncapi.content.ContentFactory getDefaultContentFactory()
        throws Exception {
        java.util.Map properties = getPropertiesFromPRB();
        if (null == properties) {
            properties = getPropertiesFromPRB();
        }
        if (null != properties) {
            String factoryClassName = (String) properties.get("ContentFactory");
            if (null != factoryClassName) {
                Class factoryClass = Class.forName(factoryClassName);        ...
                org.okiproject.okifuncapi.content.ContentFactory factory =
                    (org.okiproject.okifuncapi.content.ContentFactory) factoryClass.newInstance();
                factory.setProperties(properties);
                return factory;
            }
        }
        throw new org.okiproject.okifuncapi.content.ContentException(
            org.okiproject.okifuncapi.content.ContentException.DEFAULTCONTENTFACTORYDIDNOTLOAD);  }

    private java.util.Map properties = null;
    protected java.util.Map getProperties() { return this.properties; }
    protected void setProperties(java.util.Map properties) { this.properties = properties; }

    protected static java.util.Properties getPropertiesFromPRB() throws Exception {
        java.util.PropertyResourceBundle prb =
            (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("ContentFactory");
        if (null != prb) {
            java.util.Properties properties = new java.util.Properties();
            java.util Enumeration keys = prb.getKeys();
            while (keys.hasMoreElements()) {
                String key = (String) keys.nextElement();
                properties.setProperty(key, (String) prb.handleGetObject(key));
            }
            return properties;
        }
        throw new org.okiproject.okifuncapi.content.ContentException(
            org.okiproject.okifuncapi.content.ContentException.DEFAULTCONTENTFACTORYDIDNOTLOAD);  }

    **
     * @param props
     */
    public static org.okiproject.okifuncapi.content.ContentFactory getContentFactory(
        java.util.Map props) { return org.okiproject.okifuncapi.content.ContentFactory
        * @return org.okiproject.okifuncapi.content.ContentFactory
        * @throws Exception
        */
package org.okiproject.okifuncapi.content;

/**
* Copyright (c) 2001 Massachusetts Institute of Technology
* Permission is hereby granted, free of charge, to any
* person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions
of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
*/

public interface LAType
extends java.io.Serializable
{
    org.okiproject.okifuncapi.content.ResourceReferences getResourceReferences ();
    void setResourceReferences (org.okiproject.okifuncapi.content.ResourceReferences resourceReferences);

    org.okiproject.serviceapi.rules.Rules getRules (org.okiproject.serviceapi.rules.RuleType rulesType);
    void setRules (org.okiproject.serviceapi.rules.Rules rules);

    org.okiproject.serviceapi.rules.RuleTypes getRuleTypes ();
    //??void setRuleTypes (org.okiproject.serviceapi.rules.RuleTypes rulesTypes);

    Topics getTopics()
    void setTopics(Topics)

    Courses getCourses();
    void setCourses(Courses);

    java.util.Properties getProperties() ;
    void update(field list..)
    void delete();
    export();
    LearningAsset copy();  //** All References, Rules etc remain inherited ***/
    /** a full or true "copy" would be achieved by a copy() followed by split_off()
for each of the types */
    LearningAssets getParents();
    LearningAssets getChildren();

    Boolean searchContent(searchCriteria);
    void Split_off (org.okiproject.serviceapi.rules.RuleType);  //*** create distinct copy based on reference ***/
    change_backto reference(ruletype,id);
    setRuletypetoNull(ruletype)
    Annotate());  /* This is a special kind of link to a LA, possibly in only cert
ain subclasses */
package org.okiproject.okifuncapi.content;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology
   <p>Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,

   distribute, sublicense, and/or sell copies of the

Software, and to permit persons to whom the Software is

furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,

DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF

CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN

CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS

IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/okifuncapi/content/LATypes.java,v $
$@version@ $Revision: 1.2 $$Date: 2002/02/05 20:55:30 $ 
*/

public interface LATypes
    extends java.io.Serializable
{
    void add (org.okiproject.okifuncapi.content.LAType laType) ;
    void remove (org.okiproject.okifuncapi.content.LAType laType) ;
    void add ( void add (org.okiproject.okifuncapi.content.LAType laType) ;
    org.okiproject.okifuncapi.content.LATypesIterator iterator () ;
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * The software is provided "as is", without warranty of any
 * kind, express or implied, including but not limited to the
 * warranties of merchantability, fitness for a particular
 * purpose and noninfringement. in no event shall the
 * authors or copyright holders be liable for any claim,
 * damages or other liability, whether in an action of
 * contract, tort or otherwise, arising from, out of or in
 * connection with the software or the use or other dealings
 * in the software.
 */

public interface LATypesIterator
extends java.io.Serializable {

    /**
     * Method hasMore
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     * @return org.okiproject.okifuncapi.content.LAType
     */
    org.okiproject.okifuncapi.content.LAType next();
}
package org.okiproject.okifuncapi.content;

/**<p>Copyright (c) 2001 Massachusetts Institute of Technology<p>Permission is hereby granted, free of charge, to any<p>person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the<p>Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>Software, and to permit persons to whom the Software is<p>furnished to do so, subject to the following conditions:<p>The above copyright notice and this permission notice<p>shall be included in all copies or substantial portions of<p>the Software.<p>THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p>KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>IN THE SOFTWARE.</p>*/

public interface Learningasset extends org.okiproject.okifuncapi.content.Asset {
    java.util.Properties getProperties() ;
    void update (field list..)
    void delete ()
    export ();
    LearningAsset copy ();
    /*** All References, Rules etc remain inherited ***/
    /* a full or true "copy" would be achieved by a copy() followed by split_off() for each of the types */
    ResourceReferences getResourceReferences();
    void setResourceReferences (ResourceReferences);
    Rules getRules (RuleType);
    void setRules (Rules);
    RuleTypes listRuleTypes ();
    void split_off (RuleType); /* create distinct copy based on reference */
    /*
    change_backto reference (ruletype,id);
    setRuleGetType0Null (ruletype)
    Topics getTopics ()
    void setTopics (Topics)
    Courses getCourses ()
    void setCourses (Courses);
    LearningAssets getParents ();
    LearningAssets getChildren ()
    Boolean searchMetadata (searchCriteria);
    Boolean searchContent (searchCriteria);
    Annotate(); /* This is a special kind of link to a LA, possibly in only cert ain subclasses */
}

package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/okifuncapi/content/Learningassets.java,v $
@version# @Revision: 1.3 $
$Date: 2002/02/08 14:25:41 $
*/
public interface Learningassets
extends java.io.Serializable {

/**
 * Method add
 * @param learningasset
 */
void add(org.okiproject.okifuncapi.content.Learningasset learningasset);

/**
 * Method remove
 * @param learningasset
 */
void remove(org.okiproject.okifuncapi.content.Learningasset learningasset);

/**
 * Method iterator
 * @return org.okiproject.okifuncapi.content.LearningassetsIterator
 */
org.okiproject.okifuncapi.content.LearningassetsIterator iterator();}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface LearningassetsIterator extends java.io.Serializable {

    /**
     * Method hasMore
     *
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     *
     * @return org.okiproject.okifuncapi.content.Learningasset
     */
    org.okiproject.okifuncapi.content.Learningasset next();
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface ResourceManifest
extends java.io.Serializable {}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface ResourceReference extends org.okiproject.okifuncapi.content.Asset {
    getSize();
    getType();
    ResourceStore getResourceStore();
    getResourceAccessor(); /* path, url , external id etc */
    ResourceManifest getResourceManifest();

    update(size, ResourceType, ResourceStore, ResourceManifest);\205
    ResourceReferences searchContent(SearchCriteria);
    ResourceReferences searchMetadata(SearchCriteria);

    Rules getRules(RuleType);
    void setRules(Rules);
    RuleTypes listRuleTypes();
    void Split_off(RuleType);
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person
 * obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute,
 * sublicense, and/or sell copies of the Software, and to permit
 * persons to whom the Software is furnished to do so, subject to
 * the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions
 * of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT.  IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

import java.io.Serializable;

public interface ResourceReferences extends java.io.Serializable {

    /**
     * Method add
     * @param resourceReference
     */
    void add(org.okiproject.okifuncapi.content.ResourceReference resourceReference);

    /**
     * Method remove
     * @param resourceReference
     */
    void remove(org.okiproject.okifuncapi.content.ResourceReference resourceReference);

    /**
     * Method iterator
     * @return org.okiproject.okifuncapi.content.ResourceReferencesIterator
     */
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public interface ResourceReferencesIterator extends java.io.Serializable {
    /**
     * Method hasMore
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     * @return org.okiproject.okifuncapi.content.ResourceReference
     */
    org.okiproject.okifuncapi.content.ResourceReference next();
}
package org.okiproject.okifuncapi.content;

/**<p>   Copyright (c) 2001 Massachusetts Institute of Technology<p>   Permission is hereby granted, free of charge, to any<p>   person obtaining a copy of this software and associated<p>   documentation files (the "Software"), to deal in the<p>   Software without restriction, including without limitation<p>   the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the<p>   Software, and to permit persons to whom the Software is<p>   furnished to do so, subject to the following conditions:<p>   The above copyright notice and this permission notice<p>   shall be included in all copies or substantial portions of<p>   the Software.<p>   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY<p>   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE<p>   WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR<p>   PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE<p>   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,<p>   DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF<p>   CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN<p>   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS<p>   IN THE SOFTWARE.</p>*/

interface ResourceStore
extends java.io.Serializable{
    org.okiproject.okifuncapi.content.ResourceStoreType getType();
    setType (org.okiproject.okifuncapi.content.ResourceStoreType type);
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface ResourceStoreType
extends java.io.Serializable{}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * 
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * 
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Topic extends java.io.Serializable {
    OkiString getName ();
    void setName (OkiString name);

    String getDescription ();
    void setDescription (String description);

    OkiString getTopicMap ();
    Topic getParent ();
    void addParent (Topic);
    void removeParent (Topic);

    Topics getChildren ();
    void addChild (Topic);
    void removeChild (Topic);
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

java.io.Serializable

public interface Topics extends java.io.Serializable {

/**
 * Method add
 * @param topic
 */
void add(org.okiproject.okifuncapi.content.Topic topic);

/**
 * Method remove
 * @param topic
 */
void remove(org.okiproject.okifuncapi.content.Topic topic);

/**
 * Method iterator
 */
org.okiproject.okifuncapi.content.TopicsIterator iterator();

/**
 * Method search
 * @param searchCriteria
 */
org.okiproject.okifuncapi.content.Topics search(
SearchCriteria searchCriteria);
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 */
package org.okiproject.okifuncapi.content;

/**
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 */
package org.okiproject.okifuncapi.content;

/**
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 */
package org.okiproject.okifuncapi.content;

/**
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */
package org.okiproject.okifuncapi.content;

/**
 * $Source: /cvs/oki/org/okiproject/okifuncapi/content/TopicsIterator.java,v $
 * $Revision: 1.2 $
 * $Date: 2002/02/08 14:25:41 $
 */
package org.okiproject.okifuncapi.content;

import java.io.Serializable;

/**
 * public interface TopicsIterator
 */
public interface TopicsIterator extends java.io.Serializable {
    /**
     * Method hasMore
     *
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     *
     * @return org.okiproject.okifuncapi.content.Topic
     */
    org.okiproject.okifuncapi.content.Topic next();
}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Answer
extends org.okiproject.okifuncapi.content.Asset {}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/okifuncapi/questiontest/Answers.java,v $
$Revision: 1.2 $
$Date: 2002/02/08 14:25:41 $
*/

public interface Answers
extends java.io.Serializable {

/**
 * Method add
 *
 * @param answer
 */

void add(org.okiproject.okifuncapi.questiontest.Answer answer);

/**
 * Method remove
 *
 * @param answer
 */

void remove(org.okiproject.okifuncapi.questiontest.Answer answer);

/**
 * Method iterator
 *
 * @return org.okiproject.okifuncapi.questiontest.AnswersIterator
 */

org.okiproject.okifuncapi.questiontest.AnswersIterator iterator();}
package org.okiproject.okifuncapi.questiontest;

/**
* Copyright (c) 2001 Massachusetts Institute of Technology
* Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
* The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
* Source: /cvs/oki/org/okiproject-okifuncapi/questiontest/AnswersIterator.java,v
* @version 1.2 $Revision: 1.2 $$Date: 2002/02/08 14:25:41 $*/
public interface AnswersIterator extends java.io.Serializable {

    /**
     * Method hasMore
     *
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     *
     * @return org.okiproject.okifuncapi.questiontest.Answer
     */
    org.okiproject.okifuncapi.questiontest.Answer next();
}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public interface Gradingrule
extends java.io.Serializable {}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Gradingrules extends java.io.Serializable {
    /**
     * Method add
     *
     * @param gradingrule
     */
    void add(org.okiproject.okifuncapi.questiontest.Gradingrule gradingrule);

    /**
     * Method remove
     *
     * @param gradingrule
     */
    void remove(org.okiproject.okifuncapi.questiontest.Gradingrule gradingrule);

    /**
     * Method iterator
     *
     * @return org.okiproject.okifuncapi.questiontest.GratingrulesIterator
     */
    org.okiproject.okifuncapi.questiontest.GradingrulesIterator iterator();
}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface GradingrulesIterator
extends java.io.Serializable {

    /**
     * Method hasMore
     */
    @return Boolean
    Boolean hasMore();

    /**
     * Method next
     */
    @return org.okiproject.okifuncapi.questiontest.Gradingrule
    org.okiproject.okifuncapi.questiontest.Gradingrule next();
}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 */

public interface Measure
extends java.io.Serializable {}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
* person obtaining a copy of this software and associated
* documentation files (the "Software"), to deal in the
* Software without restriction, including without limitation
* the rights to use, copy, modify, merge, publish,
* distribute, sublicense, and/or sell copies of the
* Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
* 
* The above copyright notice and this permission notice
* shall be included in all copies or substantial portions of
* the Software.
* 
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
* KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
* WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
* PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
* DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
* CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
* CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
* IN THE SOFTWARE.
*/

public interface Measures
extends java.io.Serializable {

/**
 * Method add
 * 
 * @param measure
 */
 void add(org.okiproject.okifuncapi.questiontest.Measure measure);

/**
 * Method remove
 * 
 * @param measure
 */
 void remove(org.okiproject.okifuncapi.questiontest.Measure measure);

/**
 * Method iterator
 * 
 * @return org.okiproject.okifuncapi.questiontest.MeasuresIterator
 */
 org.okiproject.okifuncapi.questiontest.MeasuresIterator iterator();}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface MeasuresIterator extends java.io.Serializable {

/**
 * Method hasMore
 *
 * @return Boolean
 *
 */
Boolean hasMore();

/**
 * Method next
 *
 * @return org.okiproject.okifuncapi.questiontest.Measure
 *
 */
org.okiproject.okifuncapi.questiontest.Measure next();
}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$Source: /cvs/oki/org/okiproject/okifuncapi/questiontest/Question.java,v $
$Revision: 1.3 $
$Date: 2002/02/08 14:25:42 $
*/
public interface Question
extends java.io.Serializable {

/**
 * Method getTopics
 * @return org.okiproject.okifuncapi.content.Topics
 */
org.okiproject.okifuncapi.content.Topics getTopics();

/**
 * Method setTopics
 * @param topics
 */
void setTopics(org.okiproject.okifuncapi.content.Topics topics);

/**
 * Method getAnswers
 * @return org.okiproject.okifuncapi.questiontest.Answers
 */
org.okiproject.okifuncapi.questiontest.Answers getAnswers();

/**
 * Method setAnswers
 * @param answers
 */
void setAnswers(org.okiproject.okifuncapi.questiontest.Answers answers);

/**
 * Method getQuestiontype
 * @return org.okiproject.okifuncapi.questiontest.Questiontype
 */
org.okiproject.okifuncapi.questiontest.Questiontype getQuestiontype();

/**
 * Method setQuestiontype
 * @param questiontype
 */
void setQuestiontype(
org.okiproject.okifuncapi.questiontest.Questiontype questiontype);

/**
 * Method getQuestions
 * @return org.okiproject.okifuncapi.questiontest.Questions
 */
org.okiproject.okifuncapi.questiontest.Questions getQuestions();

/**
 * Method setQuestions
 * @param getQuestions
 */
void setQuestions(
org.okiproject.okifuncapi.questiontest.Questions getQuestions);

/**
 * Method search
 * @param searchCriteria
 * @return Boolean
 */
Boolean search(
org.okiproject.okifuncapi.utility.SearchCriteria searchCriteria);

/**
 * Method getWeight
 */
```java
org.okiproject.okifuncapi.questiontest.Weight getWeight();

/**
 * Method setWeight
 *
 * @param weight
 */
void setWeight(org.okiproject.okifuncapi.questiontest.Weight weight);

/**
 * Method getGradingRules
 *
 * @return org.okiproject.okifuncapi.questiontest.GradingRules
 */
org.okiproject.okifuncapi.questiontest.GradingRules getGradingRules();

/**
 * Method setGradingRules
 *
 * @param gradingRules
 */
void setGradingRules(
    org.okiproject.okifuncapi.questiontest.GradingRules gradingRules);
```
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

Source: /cvs/oki/org/okiproject/okifuncapi/questiontest/Questions.java,v $@
@version: 1.3 $@
$Date: 2002/02/08 14:25:42 $@

public interface Questions
extends java.io.Serializable {

/**
 * Method add
 *
 * @param question
 */
    void add(org.okiproject.okifuncapi.questiontest.Question question);

/**
 * Method remove
 *
 * @param question
 */
    void remove(org.okiproject.okifuncapi.questiontest.Question question);

/**
 * Method iterator
 *
 * @return org.okiproject.okifuncapi.questiontest.QuestionsIterator
 */
    org.okiproject.okifuncapi.questiontest.QuestionsIterator iterator();

/**
 * Method search
 *
 * @param searchCriteria
 * @return org.okiproject.okifuncapi.questiontest.Questions
 */
    org.okiproject.okifuncapi.questiontest.Questions search(
        org.okiproject.okifuncapi.utility.SearchCriteria searchCriteria);
}
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface QuestionsIterator
extends java.io.Serializable {

    /**
     * Method hasMore
     *
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     *
     * @return org.okiproject.okifuncapi.questiontest.Question
     */
    org.okiproject.okifuncapi.questiontest.Question next();}
/**
   * Copyright (c) 2001 Massachusetts Institute of Technology
   */
   
   The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

   public class QuestiontestException
   extends java.lang.Exception {
   public static final String UNIMPLEMENTED = "Unimplemented Questiontest method ";
   public static final String DEFAULTQUESTIONTESTFACTORYDIDNOTLOAD = "Default Questiontest factory did not load ";
   public static final String QUESTIONTESTFACTORYPROPERTIESDIDNOTLOAD = "Questiontest factory properties did not load ";

   /**
    * Constructor AuthenException
    *
    * @param message
    *
    */
   public AuthenException(String message) {
      super(message);
   }
   }
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGE OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

Source: /cvs/ok/okiproject/okifuncapi/questiontest/QuestiontestFactory.java,"src"
@version $Revision: 1.2 $ $Date: 2002/02/08 14:25:42 $
*/

public class QuestiontestFactory
  extends java.io.Serializable {

private static final org.okiproject.okifuncapi.questiontest.QuestiontestException UN
IMPLEMENTED =
    new org.okiproject.okifuncapi.questiontest.QuestiontestException(    org.okiproject.okifuncapi.questiontest.QuestiontestException.UNIMPLEMENTED);

    /**
      * Method getDefaultQuestiontestFactory
      * @return org.okiproject.okifuncapi.questiontest.QuestiontestFactory
      * @throws Exception
      */
    public static org.okiproject.okifuncapi.questiontest.QuestiontestFactory getDefaultQ
uestiontestFactory()
        throws Exception {
          java.util.Map properties = getPropertiesFromPRB();
          return getQuestiontestFactory(properties);
        }

    public static org.okiproject.okifuncapi.questiontest.QuestiontestFactory getQuestion
estionFactory(java.util.Map props)
        throws Exception {
          java.util.Map properties = props;
          if (null == properties)
            properties = getPropertiesFromPRB();
          String factoryClassName =
            (String) properties.getProperty("QuestiontestFactory");
          if (null != factoryClassName)
            Class factoryClass =
              Class.forName(factoryClassName);
          if (null != factoryClass)
            factory =
              (org.okiproject.okifuncapi.questiontest.QuestiontestFactory) factoryClass.newInstance();
          factory.setProperties(properties);
          return factory;
        }

    private java.util.Map properties = null;

    protected java.util.Map getProperties() {
      return this.properties;
    }

    protected void setProperties(java.util.Map properties) {
      this.properties = properties;
    }

    protected static java.util.Properties getPropertiesFromPRB()
        throws Exception {
          java.util.PropertyResourceBundle prb =
            (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("QuestiontestFactory");
          if (null != prb) {
            java.util.Properties properties = new java.util.Properties();
            java.util.Enumeration keys = prb.getKeys();
          while (keys.hasMoreElements()) {
            String key = (String) keys.nextElement();
            properties.setProperty(key, (String) prb.handleGetObject(key));
          }
          return properties;
        }

    protected void setProperties(java.util.Properties properties) {
      this.properties = properties;
    }

    protected void setPropertiesFromPRB() throws Exception {
      java.util.PropertyResourceBundle prb =
        (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("QuestiontestFactory");
      if (null != prb) { 
        java.util.Properties properties = new java.util.Properties();
        java.util.Enumeration keys = prb.getKeys();
        while (keys.hasMoreElements()) {
          String key = (String) keys.nextElement();
          properties.setProperty(key, (String) prb.handleGetObject(key));
        }
        return properties;
      }
throw new org.okiproject.okifuncapi.questiontest.QuestiontestException(
    org.okiproject.okifuncapi.questiontest.QuestiontestException.QUESTIONTESTFACTORY
    PROPERTIES_DID_NOT_LOAD);
package org.okiproject.okifuncapi.questiontest;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute,
 * sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */
public interface QuestionType
extends java.io.Serializable {
package org.okiproject.okifuncapi.questiontest;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

$/
$Source: /cvs/oki/org/okiproject-okifuncapi-questiontest/Weight.java,v $
$Revision: 1.2 $
$Date: 2002/02/08 14:25:42 $
*/

public interface Weight
extends java.io.Serializable {
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

/**
 * Source: /cvs/oki/org/okiproject/okifuncapi/utility/Interaction.java,v
 */

public interface Interaction
extends java.io.Serializable {

  // request
  // confirm
  // completed
  // isScheduleable ()
  // isSuspendable ()
  // isReScheduleable ()
  // begin may take a Tool
  // for triggering action
  // suspend may take a Tool
  // for triggering action
  // resume may take a Tool
  // for triggering action
  // status - may take requested, scheduled, suspended, active, completed
  // idea of joining an interaction (is this an Agent)

  /**
   * @param type
   */
  void setType(org.okiproject.okifuncapi.utility.InteractionType type);

  /**
   * @return org.okiproject.okifuncapi.utility.InteractionType
   */
  org.okiproject.okifuncapi.utility.InteractionType getType();

  /**
   * @param measures
   */
  void setMeasures(Measures measures);

  /**
   * @return Measures
   */
  Measures getMeasures();

  /**
   * @param schedule
   */
  void setSchedule(Schedule schedule);

  /**
   * @return Schedule
   */
  Schedule getSchedule();

  /**
   * @param course
   */
  void setCourse(Course course);

  /**
   * @return Course
   */
  Course getCourse();

  /**
   * @return Assets
   */
  Assets getAssets();
Assets getAssets();

/**
 * Method setAssets
 * @param assets
 */
void setAssets(Assets assets);

Roles getRoles();

/**
 * Method setRoles
 * @param roles
 */
void setRoles(Roles roles);

Rules getRules();

/**
 * Method setRules
 * @param rules
 */
void setRules(Rules rules);

Step getStep();

/**
 * Method setStep
 * @param step
 */
void setStep(Step step);
package org.okiproject.okifuncapi.utility;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
   documentation files (the "Software"), to deal in the
   Software without restriction, including without limitation
   the rights to use, copy, modify, merge, publish,
   distribute, sublicense, and/or sell copies of the
   Software, and to permit persons to whom the Software is
   furnished to do so, subject to the following conditions:
   The above copyright notice and this permission notice
   shall be included in all copies or substantial portions of
   the Software.
   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
   KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
   WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
   PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
   DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
   CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
   IN THE SOFTWARE.

Source: /cvs/oki/org/okiproject/okifuncapi/utility/InteractionType.java,v
   @version $Revision: 1.2 $ $Date: 2002/02/08 14:25:42 $
*/
public interface InteractionType
   extends java.io.Serializable {
   /**
    * Method getName
    * @return OkiString
    */
   OkiString getName();

   /**
    * Method setName
    * @param name
    */
   void setName(OkiString name);

   /**
    * Method getDescription
    * @return String
    */
   String getDescription();

   /**
    * Method setDescription
    * @param description
    */
   void setDescription(String description);

   /**
    * Method getLogs
    * @return org.okiproject.okifuncapi.logging.Logs
    */
   org.okiproject.okifuncapi.logging.Logs getLogs();

   /**
    * Method setLogs
    * @param logs
    */
   void setLogs(org.okiproject.okifuncapi.logging.Logs logs);

   /**
    * Method getRules
    * @return org.okiproject.serviceapi.rules.Rules
    */
   org.okiproject.serviceapi.rules.Rules getRules();

   /**
    * Method setRules
    * @param rules
    */
   void setRules(org.okiproject.serviceapi.rules.Rules rules);

   // possibly Rulestypes

   /**
    * Method getAssettypes
    * @return org.okiproject.okifuncapi.content.Assettypes
    */
   org.okiproject.okifuncapi.content.Assettypes getAssettypes();

   /**
    * Method setAssettypes
    * @param assettypes
    */
   void setAssettypes(org.okiproject.okifuncapi.content.Assettypes assettypes);
// possibly Tools and Tooltypes
}

package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Interactions extends java.io.Serializable {

/**
 * Method add
 *
 * @param interaction
 */
    void add(org.okiproject.okifuncapi.utility.Interaction interaction);

/**
 * Method remove
 *
 * @param interaction
 */
    void remove(org.okiproject.okifuncapi.utility.Interaction interaction);

/**
 * Method iterator
 *
 * @return org.okiproject.okifuncapi.utility.InteractionsIterator
 */
    org.okiproject.okifuncapi.utility.InteractionsIterator iterator();}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology

 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute,
 * sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:

 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.

 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.

 */

Source: /cvs/oki/org/okiproject/okifuncapi/utility/InteractionsIterator.java,v $ @version $ @Revision: 1.2 $ @date: 2002/02/08 14:25:42 $ */

public interface InteractionsIterator
        extends java.io.Serializable {
            /**
             * Method hasMore
             *
             * @return Boolean
             */
            Boolean hasMore();

            /**
             * Method next
             *
             * @return org.okiproject.okifuncapi.utility.Interaction
             */
            org.okiproject.okifuncapi.utility.Interaction next();
        }
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public interface Message
extends java.io.Serializable {}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public interface Messagequeue
extends java.io.Serializable {

/**
 * Method put
 *
 * @param message
 */
void put(org.okiproject.okifuncapi.utility.Message message);

/**
 * Method get
 *
 * @return org.okiproject.okifuncapi.utility.Message
 */
org.okiproject.okifuncapi.utility.Message get();

/**
 * Method get
 *
 * @param queueIndex
 * @return org.okiproject.okifuncapi.utility.Message
 */
org.okiproject.okifuncapi.utility.Message get(Integer queueIndex);

/**
 * Method peek
 *
 * @return org.okiproject.okifuncapi.utility.Message
 */
org.okiproject.okifuncapi.utility.Message peek();

/**
 * Method search
 *
 * @param message
 * @return Integer
 */
Integer search(org.okiproject.okifuncapi.utility.Message message);
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Schedule
    extends java.io.Serializable {

    /**
     * Method getDates
     * @return Dates
     */
    Dates getDates();

    /**
     * Method setDates
     */
    void setDates();

    /**
     * Method getLocations
     * @return Locations
     */
    Locations getLocations();

    /**
     * Method setLocations
     * @param locations
     */
    void setLocations(Locations locations);

    /**
     * Method getInteractions
     * @return Interactions
     */
    Interactions getInteractions();

    /**
     * Method setInteractions
     * @param interactions
     */
    void setInteractions(Interactions interactions);

    /**
     * Method getSelf
     */
    void getSelf();

    /**
     * Method setSelf
     */
    void getSelf();
}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * 
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * 
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface SearchCriteria
extends java.io.Serializable {}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Tool
    extends org.okiproject.okifuncapi.content.Asset
{
    void invoke();
    Ruletypes listControlRuleTypes();
    Ruletypes listInterpretRuleTypes();
    Vocabs listVocabs(Ruletype);
    LearningAssetTypes listLearningAssetTypes();
    Logs getLogs();
    void setLogs(Logs);
}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface Tools
        extends java.io.Serializable {

    /**
     * Method add
     * @param tool
     */
    void add(org.okiproject.okifuncapi.utility.Tool tool);

    /**
     * Method remove
     * @param tool
     */
    void remove(org.okiproject.okifuncapi.utility.Tool tool);

    /**
     * Method iterator
     * @return org.okiproject.okifuncapi.utility.ToolsIterator
     */
    org.okiproject.okifuncapi.utility.ToolsIterator iterator();}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */
public interface ToolsIterator
extends java.io.Serializable {

/**
 * Method hasMore
 *
 * @return Boolean
 */
Boolean hasMore();

/**
 * Method next
 *
 * @return org.okiproject.okifuncapi.utility.Tool
 */
org.okiproject.okifuncapi.utility.Tool next();
}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
<p>
The above copyright notice and this permission notice shall be included in all copies or substantial portions of
the Software.
<p>
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.
<p>
Source: /cvs/oki/org/okiproject/okifuncapi/utility/Tooltype.java,v $@
@version@ $Revision: 1.2 $$Date: 2002/02/08 14:25:42 $@
*/

public interface Tooltype
extends java.io.Serializable {}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */
public interface Tooltypes extends java.io.Serializable {

/**
 * Method add
 * @param tooltype
 */
void add(org.okiproject.okifuncapi.utility.Tooltype tooltype);

/**
 * Method remove
 * @param tooltype
 */
void remove(org.okiproject.okifuncapi.utility.Tooltype tooltype);

/**
 * Method iterator
 * @return org.okiproject.okifuncapi.utility.TooltypesIterator
 */
org.okiproject.okifuncapi.utility.TooltypesIterator iterator();}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy
 * of this software and associated documentation files (the "Software"), to deal in
 * the Software without restriction, including without limitation the rights to use,
 * copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is furnished to do so,
 * subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all
 * copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
 * INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
 * PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
 * HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
 * OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
 * SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

public interface TooltypesIterator
    extends java.io.Serializable {
        /**
         * Method hasMore
         * @return Boolean
         */
        Boolean hasMore();

        /**
         * Method next
         * @return org.okiproject.okifuncapi.utility.Tooltype
         */
        org.okiproject.okifuncapi.utility.Tooltype next();}
package org.okiproject.okifuncapi.utility;

/**
   Copyright (c) 2001 Massachusetts Institute of Technology
   Permission is hereby granted, free of charge, to any
   person obtaining a copy of this software and associated
documentation files (the "Software"), to deal in the
Software without restriction, including without limitation
the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the
Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice
shall be included in all copies or substantial portions of
the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
IN THE SOFTWARE.

/**
Source: /cvs/oki/okiproject/okifuncapi/utility/UtilityException.java,v $ @version@ $Revision: 1.2 $ $Date: 2002/02/08 14:25:42 $ */
public class UtilityException
extends java.lang.Exception {

public static final String UNIMPLEMENTED =
"Unimplemented Utility method ";

public static final String DEFAULTUTILITYFACTORYDIDNOTLOAD =
"Default Utility factory did not load ";

public static final String UTILITYFACTORYPROPERTIESDIDNOTLOAD =
"Utility factory properties did not load ";

/**
 * Constructor UtilityException
 * @param message
 */
public UtilityException(String message) {
    super(message);
}
package org.okiproject.okifuncapi.utility;

/** Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

S@version@ $Revision: 1.2 $$Date: 2002/02/08 14:25:42 $*/

import java.util.Properties;

private static final org.okiproject.okifuncapi.utility.UtilityException UNIMPLEMENTED;

/** * Method getDefaultUtilityFactory * @return org.okiproject.okifuncapi.utility.UtilityFactory * @throws Exception */

public static org.okiproject.okifuncapi.utility.UtilityFactory getDefaultUtilityFactory() throws Exception {
    java.util.Map properties = getPropertiesFromPRB();
    if (null == properties) {
        properties = getPropertiesFromPRB();
    }
    if (null != properties) {
        String factoryClassName = properties.getProperty("UtilityFactory");
        if (null != factoryClassName) {
            Class factoryClass = Class.forName(factoryClassName);
            factory = (org.okiproject.okifuncapi.utility.UtilityFactory) factoryClass.newInstance();
            factory.setProperties(properties);
            return factory;
        }
    }
    throw new org.okiproject.okifuncapi.utility.UtilityException("DEFAULTUTILITYFACTORYDIDNOTLOAD");
}

private java.util.Map properties = null;

protected java.util.Map getProperties() {
    return this.properties;
}

protected void setProperties(java.util.Map properties) {
    this.properties = properties;
}

protected static java.util.Properties getPropertiesFromPRB() throws Exception {
    java.util.Properties properties = new java.util.Properties();
    java.util.Properties prb = (java.util.PropertyResourceBundle) java.util.ResourceBundle.getBundle("UtilityFactory");
    if (null != prb) {
        java.util.Properties prp = new java.util.Properties();
        java.util.Enumeration keys = prb.getKeys();
        while (keys.hasMoreElements()) {
            String key = (String) keys.nextElement();
            properties.setProperty(key, (String) prb.handleGetObject(key));
        }
        return properties;
    }
    throw new org.okiproject.okifuncapi.utility.UtilityException("DEFAULTUTILITYFACTORYDIDNOTLOAD");
}
org.okiproject.okifuncapi.utility.UtilityFactory.java

org.okiproject.okifuncapi.utility.UtilityException.UTILITYFACTORYPROPERTIESDIDNOTLOAD;
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 */

public interface Vocab
extends java.io.Serializable {

/**
 * Method add
 * @param str
 */
 void add(OkiString str);

/**
 * Method remove
 * @param str
 */
 void remove(OkiString str);

/**
 * Method iterator
 * @return org.okiproject.okifuncapi.utility.VocabIterator
 */
 org.okiproject.okifuncapi.utility.VocabIterator iterator();
}
package org.okiproject.okifuncapi.content;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 * 
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * 
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 *
**/@
public interface VocabIterator
extends java.io.Serializable {
    /**
     * Method hasMore
     * @return Boolean
     */
    Boolean hasMore();

    /**
     * Method next
     * @return OkiString
     */
    OkiString next();}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any
 * person obtaining a copy of this software and associated
 * documentation files (the "Software"), to deal in the
 * Software without restriction, including without limitation
 * the rights to use, copy, modify, merge, publish,
 * distribute, sublicense, and/or sell copies of the
 * Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 * The above copyright notice and this permission notice
 * shall be included in all copies or substantial portions of
 * the Software.
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY
 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
 * WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
 * PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
 * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 */

public interface Work
    extends java.io.Serializable {}
package org.okiproject.okifuncapi.utility;

/**
 * Copyright (c) 2001 Massachusetts Institute of Technology
 * Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
 * 
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

Source: /cvs/oki/org/okiproject/okifuncapi/utility/Workqueue.java,v $<version> $Revision: 1.3 $ $Date: 2002/02/08 14:25:42 $ */

public interface Workqueue extends java.io.Serializable {

/**
 * Method put
 *
 * @param work
 */
 void put(org.okiproject.okifuncapi.utility.Work work);

/**
 * Method get
 *
 * @return org.okiproject.okifuncapi.utility.Work
 */
 org.okiproject.okifuncapi.utility.Work get();

/**
 * Method get
 *
 * @param queueIndex
 * @return org.okiproject.okifuncapi.utility.Work
 */
 org.okiproject.okifuncapi.utility.Work get(Integer queueIndex);

/**
 * Method peek
 *
 * @return org.okiproject.okifuncapi.utility.Work
 */
 org.okiproject.okifuncapi.utility.Work peek();

/**
 * Method search
 *
 * @param work
 * @return Integer
 */
 Integer search(org.okiproject.okifuncapi.utility.Work work);
}