LexEVS 6.0 CTS2 Administration 3 - Notification API

Contents of this Page

CTS2 Links for LexEVS 6.0

- CTS2 API Main Page
- Programmer's Guide Main Page
- LexEVS 6.0 Main Page
- LexEVS Current Release

Introduction

This document is intended for LexEVS developers, explaining how to use the CTS2 Notification API.

Notification API

The Notification API is accessed via the interface: **org.lexevs.cts2.admin.NotificationAdminOperation** JavaDoc

```
/*
 * Copyright: (c) 2004-2009 Mayo Foundation for Medical Education and Research
 * (MFMER). All rights reserved. MAYO, MAYO CLINIC, and the triple-shield Mayo
 * logo are trademarks and service marks of MFMER.
 *
 * Except as contained in the copyright notice above, or as used to identify
 * MFMER as the author of this software, the trade names, trademarks, service
 * marks, or product names of the copyright holder shall not be used in
 * advertising, promotion or otherwise in connection with this software without
 * prior written authorization of the copyright holder.
 *
 * Licensed under the Eclipse Public License, Version 1.0 (the "License"); you
 * may not use this file except in compliance with the License. You may obtain a
 * copy of the License at
 *
 * http://www.eclipse.org/legal/epl-v10.html
 */
package org.lexevs.cts2.admin;

import org.lexevs.cts2.exception.admin.NotificationNotRegisteredException;
import org.lexevs.dao.database.service.event.DatabaseServiceEventListener;

/**
 * The Interface NotificationAdminOperation controls the registration and maintenence of Content
 * Notification Listeners.
 */
public interface NotificationAdminOperation {

        /**
         * The Enum NotificationStatus.
         *
         * SUSPEND: Notifications will cease being processed until further notice.
         * REINSTATE: Notifications will resume processing.
         * REMOVE: The Notification will be removed, and cannot be REINSTATE'd.
         */
        public enum NotificationStatus {
                        SUSPEND,
                        REINSTATE,
```

```
                        REMOVE}

        /**
         * Register the given Notification Listener in the system.
         *
         * @param listener
         *                 The Notification Listener
         *
         * @return
         *                 The Notification Identifier: The unique identifier of the particular Notification
         *                 in the system.
         */
        public String registerForNotification(DatabaseServiceEventListener listener);

        /**
         * Replaces a Notification Listener with an Updated Notification Listener
         *
         * @param notificationId
         *                 The Notification Identifier of the Notification to be updated.
         * @param listener the listener
         */
        public void updateNotificationRegistration(String notificationId, DatabaseServiceEventListener listener)
                throws NotificationNotRegisteredException;

        /**
         * Update the Notification Registration Status of a Notification Listener.
         *
         * @param status
         *                 The new Notification Registration Status
         * @param notificationId
         *                 The Notification Identifier of the Notification to be updated.
         */
        public void updateNotificationRegistrationStatus(String notificationId, NotificationStatus status)
                throws NotificationNotRegisteredException;
}
```

Listening For Events

A class implementing the interface **org.lexevs.dao.database.service.event.DatabaseServiceEventListener** may be used to listen for various content-related events.

A default listener is provided in the class **org.lexevs.dao.database.service.listener.DefaultServiceEventListener**. The listener methods of this class are intended to be overridden by subclasses, enabling the subclass to choose which events to process.

Example Usecases

Various usecases exist for using Notifications, including:

- Email notices
- Custom security implementations
- Additional change logs or auditing
- etc...

> **Note**
>
> LexEVS does not supply any of the above Notification implementations, it merely supplies the API to register/unregister the listeners, and the event API itself.

Examples

- Email when a CodingScheme is updated.
  Sends an email to a defined user when any CodingScheme is updated in the system.

```
package org.lexevs.alert;

import java.util.Properties;

import javax.mail.Message;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

import org.LexGrid.LexBIG.Extensions.Generic.GenericExtension;
import org.lexevs.dao.database.service.event.DatabaseServiceEventListener;
import org.lexevs.dao.database.service.event.codingscheme.CodingSchemeUpdateEvent;
import org.lexevs.dao.database.service.listener.DefaultServiceEventListener;

public class MailAlertExtension extends DefaultServiceEventListener implements GenericExtension,
DatabaseServiceEventListener{

	private static final long serialVersionUID = 6752732957279346146L;

	public static void main(String[] args) {
		MailAlertExtension ext = new MailAlertExtension();
		ext.onCodingSchemeUpdate(null);
	}

	public boolean onCodingSchemeUpdate(CodingSchemeUpdateEvent event) {
		try {

			Properties props = System.getProperties();
			props.put("mail.smtp.host", "your.host.org");
			props.put("mail.smtp.port", "25");

			Session session = Session.getDefaultInstance(props,null);

			Message msg = new MimeMessage(session);

			msg.setFrom(new InternetAddress("user@your.host.org"));

			msg.setRecipients(Message.RecipientType.TO, InternetAddress.parse("NotifiedUser",
false));

			msg.setSubject("Coding Scheme Change");
			msg.setText("Coding Scheme:" + event.getOriginalCodingScheme().getCodingSchemeURI() + "
has changed.");

			Transport.send(msg);
		} catch(Exception e) {
			throw new RuntimeException(e);
		}

		return true;
	}

	public String getDescription() {
		return "An Email alerting Extension";
	}

	public String getName() {
		return "MailAlertExtension";
	}

	public String getProvider() {
		return "LexEVS Team";
	}

	public String getVersion() {
		return "1.0";
	}
}
```