Oracle® Multimedia

JSP Tag Library Guide

11*g* Release 2 (11.2)

E10621-01

September 2009

Oracle Multimedia JSP Tag Library is an extension of Oracle Multimedia Servlets and JSP Java API that simplifies retrieving and uploading media data from and to Oracle Database in multimedia JSP Web applications.

ORACLE®

Oracle Multimedia JSP Tag Library Guide, 11*g* Release 2 (11.2)

E10621-01

Primary Author: Sue Pelski

Contributors: Melliyal Annamalai, Fengting Chen, Dong Lin, Sue Mavris, Susan Shepard, Manjari Yalavarthy

Contents

List of Examples

List of Tables

Preface

This guide describes how to use the Oracle Multimedia JSP Tag Library.

In Oracle Database 11*g* Release 1 (11.1), the name Oracle *inter*Media was changed to Oracle Multimedia. The feature remains the same, only the name has changed. References to Oracle *inter*Media were replaced with Oracle Multimedia, however some references to Oracle *inter*Media or *inter*Media might still appear in graphical user interfaces, code examples, and related documents in the Documentation Library for Oracle Database 11*g*.

Audience

This guide is for experienced application developers who want to develop JSP-based multimedia Web applications of a low to medium level of complexity. It is also for application developers who want to rapidly create multimedia application prototypes and for authors of JavaServer Pages who have some experience writing Java applications.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at http://www.fcc.gov/cgb/consumerfacts/trs.html, and a list of phone numbers is available at http://www.fcc.gov/cgb/dro/trsphonebk.html.

Related Documents

Note: For details about installation and configuration, and for information added after the release of this guide, see the online `README.txt` file included in this kit.

See your operating system-specific installation guide for more information.

For more information about Oracle Multimedia, see the following documents in the Oracle Database Online Documentation Library:

- *Oracle Multimedia Reference*
- *Oracle Multimedia User's Guide*

For reference information about Oracle Multimedia Java classes in Javadoc format, see the following Oracle API documentation (also known as Javadoc) in the Oracle Database Online Documentation Library:

- *Oracle Multimedia Java API Reference*
- *Oracle Multimedia Servlets and JSP Java API Reference*

For more information about Java, including information about Java Advanced Imaging (JAI) and the JavaServer Pages Standard Tag Library (JSTL), see the API documentation provided by Sun Microsystems.

For more information about using Oracle Multimedia in a development environment, see these Oracle resources:

- Oracle Database Online Documentation Library
- Oracle Fusion Middleware Online Documentation Library

Conventions

The following text conventions are used in this guide:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Syntax Descriptions

Syntax descriptions are provided in this book for various Java command-line constructs in graphic form or Backus Naur Form (BNF). See the Oracle Database Online Documentation Library and the API documentation provided by Sun Microsystems for information about how to interpret these descriptions.

What's New in Oracle Multimedia JSP Tag Library

This document summarizes any new features introduced in the current release.

New Features for Oracle Database 11*g* Release 2

Oracle Database 11*g* Release 2 (11.2) includes *no* new features for Oracle Multimedia JSP Tag Library.

1

Introduction to Oracle Multimedia JSP Tag Library

Oracle Multimedia provides a custom JavaServer Pages (JSP) tag library that lets users easily generate multimedia HTML tags in JavaServer Pages, and upload multimedia data into Oracle Multimedia objects.

This chapter briefly describes selected Oracle Multimedia and Java concepts to show how the Oracle Multimedia JSP Tag Library fits into a JSP-based multimedia Web application.

This chapter includes the following sections:

1.1 Oracle Multimedia Overview

Oracle Multimedia enables the Oracle Database software to store, retrieve, manage, and manipulate images, audio, video, and other media data, while integrating it with other enterprise information.

Specifically, Oracle Multimedia supports media storage, retrieval, management, and manipulation of media data managed by Oracle Database and stored in one of the following: binary large objects, file-based large objects, URLs that contain media data, and user-defined storage. Oracle Multimedia media storage and retrieval services enhance the management of Web content.

Oracle Multimedia is accessible to applications through both relational and object interfaces. Database applications written in Java, C++, or traditional 3GLs can interact with Oracle Multimedia through modern class library interfaces, or PL/SQL and Oracle Call Interface.

Oracle Multimedia uses object types that are similar to Java or C++ classes to describe media data. These Oracle Multimedia object types are called ORDAudio, ORDDoc, ORDImage, ORDVideo, and ORDDicom. Oracle Multimedia objects have a common media data storage model.

Note: Currently, Oracle Multimedia JSP Tag Library does not include support for the ORDDicom object type.

See *Oracle Multimedia DICOM Developer's Guide* for more information about the ORDDicom object type.

Oracle Multimedia also provides Java APIs to support Java application development, including the use of Java servlets and JavaServer Pages (see Section 1.2.2 and Section 1.2.3, respectively).

Oracle Multimedia Java API enables users to write Java applications using Oracle Multimedia objects. Oracle Multimedia lets you store your media information in a database table, retrieve it from the table, and manipulate it. Oracle Multimedia Java API lets you write your own Java applications to use, manipulate, and modify media data stored in the database. Oracle Multimedia Java API lets an application retrieve an object from a result set and manipulate the contents of the object (see Section 1.1.1).

Oracle Multimedia Servlets and JSP Java API facilitates retrieving and uploading media data from and to the database (see Section 1.1.2).

See *Oracle Multimedia User's Guide* and *Oracle Multimedia Reference* for additional information about Oracle Multimedia and Oracle Multimedia objects. See *Oracle Multimedia Java API Reference* and *Oracle Multimedia Servlets and JSP Java API Reference* for information about using Oracle Multimedia in Java applications.

1.1.1 Media Retrieval Concepts

Oracle Multimedia Servlets and JSP Java API uses the OrdHttpResponseHandler class to retrieve media data from an Oracle database and deliver it to a browser or other HTTP client from a Java servlet or JSP page.

Oracle Multimedia JSP Tag Library provides media retrieval tags, which JSP developers can use to generate complete HTML multimedia tags or create multimedia retrieval URLs for inclusion in the customized use of an HTML multimedia tag. The media retrieval tags are listed and briefly described in Table 1–1.

Table 1–1 Oracle Multimedia JSP Tag Library Media Retrieval Tags

JSP Tag	Summary Description
embedAudio	Builds an HTML <OBJECT> tag and <EMBED> tag to embed an audio object in a page.
embedImage	Builds an HTML <IMG> tag to embed an image in a page.
embedVideo	Builds an HTML <OBJECT> tag and <EMBED> tag to embed a video object in a page.
mediaUrl	Generates a media retrieval URL object that can be used in the tag body.

See General Format for Media Retrieval Tags and Media Retrieval URL Format for additional reference information.

1.1.2 Media Upload Concepts

Oracle Multimedia Servlets and JSP Java API uses the OrdHttpUploadFile class to facilitate the handling of uploaded media files. This class provides a simple application programming interface (API) that applications call to load media data into the database.

File uploading using HTML forms encodes form data and uploaded files in POST requests using the multipart/form-data format. The OrdHttpUploadFormData class facilitates the processing of such requests by parsing the POST data and making the contents of regular form fields and the contents of uploaded files readily accessible to a Java servlet or JSP page.

Oracle Multimedia JSP Tag Library provides media upload tags, which facilitate the development of multimedia applications that upload media data into the database. The media upload tags are listed and briefly described in Table 1–2:

Table 1–2 Oracle Multimedia JSP Tag Library Media Upload Tags

JSP Tag	Summary Description
storeMedia	Operates within the body of the uploadFormData tag by implicitly using the form-data object created by the uploadFormData tag to load the uploaded media data from the HTML form into the OrdImage, OrdAudio, OrdVideo, or OrdDoc object in the specified table, row, and column in the database.
uploadFile	Operates within the uploadFormData tag to provide access to uploaded file information by using the object created by the uploadFormData tag implicitly.
uploadFormData	Parses the multipart/form-data HTTP request to provide access to text-based form parameters and the contents of uploaded files transmitted from a browser to a Web server.

1.2 Java Concepts

Java is an object-oriented programming language developed by Sun Microsystems. Java programs can run on any computer platform that includes a Java Virtual Machine (JVM). The JVM is a special software environment that compiles the code. It provides the software that the Java program uses to perform tasks including collecting user or system input, displaying information, and allocating memory. Java can be used to develop small Web applications or large network applications.

For more information about Java, see the Web site for Sun Microsystems at

`http://java.sun.com`

1.2.1 J2EE

Java2 Enterprise Edition (J2EE) is an integrated development environment developed by Sun Microsystems that extends the capabilities of the earlier Java2 Standard Edition (J2SE) platform. Among other features, J2EE supports Java servlets, JSP pages, and XML.

1.2.2 Java Servlets

Java servlets are Java programs that let Web developers extend and enhance the features of a Web server. Java servlets encourage modular programming using the latest features of the Java language, including Java classes and streams. Java servlets can access the latest Java APIs and a collection of calls specific to HTTP. Java servlets are supported on any platform that has a JVM, and a Web server that supports servlets.

Most Java servlets receive requests from a client, generate dynamic HTML text in response, and then send that text back to the client to be displayed by the Web browser. Java servlets can also generate XML text to encapsulate data and send the data to the client or to other parts of the application. Java servlets are often used to create interactive applications.

1.2.3 JavaServer Pages

JavaServer Pages (JSP) technology is an extension of Java servlet technology. It enables users to include portions of Java code in an HTML page or another type of document, such as an XML file.

In a typical JSP page, form is separate from function. The form or layout of information is determined by the JSP page author, while other functions, such as database access, are handled by calls to JavaBeans or by other mechanisms.

1.2.4 JSP Tag Libraries

All JSP tag libraries extend JSP technology. Typically, JSP tag libraries include specific, modular functions encapsulated within a set of tags that can be used by any JSP page.

Custom JSP tag libraries provide the capability to include Java functions without having to enter Java code specifically. Each JSP tag library defines a specialized sublanguage, thus enabling a more natural use of those Java functions with JSP pages. For example, Oracle Containers for Java EE (OC4J) provides the JSP Markup Language (JML) Tag Library. Along with other features, JML enables application developers to write JSP pages that include conditional logic and loop constructs without having to use the syntax for Java code scripts directly.

1.3 Choosing the Appropriate Oracle Multimedia Java Interface

Deciding whether to use Oracle Multimedia Servlets and JSP Java API or the Oracle Multimedia JSP Tag Library depends on the type of application you want to develop and your level of experience with Java programming.

Java servlets and JSP pages offer flexibility by enabling you to customize the application output. This customization requires a moderate level of skill with the Java programming language, and with servlet and JSP technology. Thus, servlets and JSP pages are best suited for complex applications developed by reasonably experienced Java programmers.

The tags provided by the Oracle Multimedia JSP Tag Library offer speed and ease of use, enabling you to write simple Java applications requiring little or no customization. These JSP tags are appropriate for less experienced Java programmers who want to develop common applications quickly, using limited Java code.

2

Oracle Multimedia JSP Tag Library Examples

This chapter provides full-length code examples of user-defined JSP pages using the Oracle Multimedia JSP Tag Library with the Oracle Multimedia JSP Tag Library Photograph Album Demonstration sample application.

The material in this chapter assumes prior knowledge of SQL, PL/SQL, Java Database Connectivity (JDBC), JSP, HTML, and XML.

This chapter includes the following sections:

> **Note:** This chapter contains examples of Java, SQL, and HTML code. This code might not match the code in the files shipped as a sample application on OTN. To run an example on your system, use the files provided with the Oracle Multimedia JSP Tag Library software on OTN; do not attempt to compile and run the code presented in this chapter.
>
> You can download the Oracle Multimedia JSP Tag Library software from the Oracle Multimedia Software section of the Oracle Technology Network Web site at
>
> `http://www.oracle.com/technology/products/multimedia/`

2.1 Table Definition for Oracle Multimedia JSP Tag Library Examples

The media retrieval and media upload examples in this chapter use the database table named `photos`, which is defined as follows:

```
photos( id      NUMBER UNIQUE NOT NULL,
        description VARCHAR2(40) NOT NULL,
        location    VARCHAR2(40),
        image       ORDSYS.ORDIMAGE,
        thumb       ORDSYS.ORDIMAGE);
```

For more information about the `photos` table, see *Oracle Multimedia User's Guide*.

The media retrieval and media upload examples also use tags from the JavaServer Pages Standard Tag Library (JSTL) to access the database. For more information about JSTL, see the Sun Microsystems Web site at

http://java.sun.com/

2.2 Media Retrieval Example

This section describes the PhotoAlbum.jsp file, which is one component of a sample JSP application that uses tags from the Oracle Multimedia JSP Tag Library to retrieve media data from the database and deliver it to a browser. The browser then displays the media in a simple photograph album application. Example 2–1 shows how to use the JSP tag embedImage to retrieve and deliver the media data.

The PhotoAlbum.jsp file generates the HTML code that displays the contents of the database table named photos, including the contents of the description, location, and thumb columns. The contents of the thumb column are displayed as thumbnail images that link to the full-size images, which are stored in the image column in the photos table. From the browser, users can click a thumbnail image to view the full-size image.

In Example 2–1, appropriate statements are highlighted in bold to show where the main actions occur.

Example 2–1 Media Retrieval (PhotoAlbum.jsp)

```
<%@ page language="java" %>
<%@ page isELIgnored ="false" %>
<%@ taglib prefix="ord"
  uri="http://xmlns.oracle.com/jsp/ord/multimedia-taglib.tld" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>

<%-- HTML header --%>
<HTML LANG="EN">
<HEAD>
<TITLE>Oracle Multimedia JSP Tag Library Photograph Album Demonstration</TITLE>
</HEAD>

<BODY>

<%-- Page heading --%>
<TABLE BORDER="0" WIDTH="100%">
  <TR>
    <TD COLSPAN="2" BGCOLOR="#F7F7E7" ALIGN="CENTER"><FONT SIZE="+2">
      Oracle Multimedia JSP Tag Library Photograph Album Demonstration</FONT>
    </TD>
  </TR>
</TABLE>

<%-- Display the thumbnail images in a table; output the table headers. --%>
<P>
<TABLE BORDER="1" CELLPADDING="3" CELLSPACING="0" WIDTH="100%"
  SUMMARY="Table of thumbnail images">
  <TR BGCOLOR="#336699">
    <TH id="description"><FONT COLOR="#FFFFFF">Description</FONT></TH>
    <TH id="location"><FONT COLOR="#FFFFFF">Location</FONT></TH>
    <TH id="image"><FONT COLOR="#FFFFFF">Image</FONT></TH>
  </TR>
```

```
<%-- Display all the entries --%>
<sql:query  var="photoQuery" dataSource="jdbc/OracleDS">
    SELECT Id, Description, Location from Photos order by Description
</sql:query>
<c:forEach var="row" items = "${photoQuery.rows}">
  <c:set var="id" value="${row.ID}"/>
  <c:set var="description" value="${fn:escapeXml(row.DESCRIPTION)}"/>
  <c:set var="location" value="${fn:escapeXml(row.LOCATION)}"/>

  <%-- Display an entry in the album --%>
  <TR>
    <TD HEADERS="description"> ${description} </TD>
    <TD HEADERS="location"> ${location}    </TD>
    <TD HEADERS="image">
              <A HREF="PhotoAlbumEntryViewer.jsp?id=${id}">
                   <%-- Use thumbnail image for anchor tag. --%>
                   <ord:embedImage dataSourceName="jdbc/OracleDS"
                                   table = "Photos"
                                   column = "Thumb"
                                   key = "${id} "
                                   keyColumn = "Id"
                                   retrievalPath="OrdGetMediaJsp.jsp"
                                   alt = "${description}"
                                   border="1" />

              </A>
    </TD>
  </TR>

</c:forEach>
    <%-- Display a message if the album is empty; otherwise, output a
         message telling the user how to view the full-size entry. --%>
  <TR>
      <TD SCOPE="col" COLSPAN="3" ALIGN="CENTER"><FONT COLOR="#336699"><B><I>
      <c:choose>
        <c:when test="${photoQuery.rowCount==0}">
          The photograph album is empty
        </c:when>
        <c:otherwise>
          Select the thumbnail image to view the full-size image
        </c:otherwise>
      </c:choose>
      </I></B></FONT></TD>
    </TR>
<%-- Finish the table --%>
</TABLE>
</P>

<%-- Output a link to the upload form --%>
<P>
<TABLE WIDTH="100%">
  <TR BGCOLOR="#F7F7E7">
    <TD COLSPAN="3" ALIGN="CENTER">
      <A HREF="PhotoAlbumUploadForm.jsp">Upload new photograph</A>
    </TD>
  </TR>
</TABLE>
</P>
```

```
</BODY>
</HTML>
```

The bolded Java, SQL, and HTML statements, in order of appearance in Example 2–1, perform the following operations:

- Provide the required prefix and uri attributes in the JSP directives. The value of the prefix attribute specifies the XML namespace identifier, which must be inserted before each occurrence of the library's tags in the JSP page. The value of the uri attribute indicates the location of the tag library descriptor (TLD) file for the specified tag library. In this example, the value `"ord"` specifies the XML namespace identifier for the Oracle Multimedia JSP Tag Library. And, the prefix attribute values `"sql"`, `"c"`, and `"fn"` specify the namespace identifiers for the JSTL sql, core, and functions tag libraries, respectively.
- Use an HTML table to display the entries of the `photos` table in the database.
- Use the JSTL sql tag to open a database connection and perform a query on the `photos` table.
- Use JSTL core tags to loop over the result set to retrieve data.
- Use the JSP tag embedImage to generate the HTML <IMG> tag that displays the `thumb` column of the `photos` table (see embedImage for information about the HTML output). The HTML <A HREF> tag uses the JSP tag embedImage as the link anchor.

2.3 Media Upload Example

This section describes the `PhotoAlbumInsertPhoto.jsp` file, which is one component of a sample JSP application that uses tags from the Oracle Multimedia JSP Tag Library to upload media files into a database. Example 2–2 shows how to use the JSP tags uploadFormData, uploadFile, and storeMedia to upload the media files.

In Example 2–2, appropriate statements are highlighted in bold to show where the main actions occur.

Example 2–2 Media Upload (PhotoAlbumInsertPhoto.jsp)

```
<%@ page language="java" %>
<%@ page isELIgnored ="false" %>
<%@ taglib prefix="ord"
  uri="http://xmlns.oracle.com/jsp/ord/multimedia-taglib.tld" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>



<%-- Upload the data into the database --%>
<ord:uploadFormData formDataId = "fd">

  <ord:uploadFile
         parameter = "photograph"
         fullFileName = "ffName"
         shortFileName = "sfName"
         length = "fLength" >

    <%-- //check if an image file is specified.  --%>
    <c:choose>
      <c:when test="${(ffName==null) or empty(ffName)}">
```

```
        <c:redirect
url="PhotoAlbumUploadForm.jsp?error=Please+supply+a+file+name."/>
      </c:when>
      <c:when test="${fLength eq 0}">
        <c:redirect
url="PhotoAlbumUploadForm.jsp?error=Please+supply+a+valid+image+file."/>
      </c:when>

      <c:otherwise>

        <%
        // Use file name as description if not supplied.
        String description = fd.getParameter("description");
        String location = fd.getParameter("location");
        if ( description == null || description.length() == 0 )
        {
            description = "Image from file: " + sfName + ".";
            if(description.length() > 40)
            {
              description = description.substring(0, 40);
            }
        }
        }

        java.util.Vector otherValuesVector = new java.util.Vector();
        otherValuesVector.add(description);
        otherValuesVector.add(location);
        %>

        <sql:setDataSource dataSource = "jdbc/OracleDS"
                           var="myDS" scope="page"/>

        <sql:transaction dataSource="${myDS}">

          <%-- Get the next id in the database sequence --%>
          <sql:query var="idQuery" >
              SELECT photos_sequence.nextval from Dual
          </sql:query>

           <%-- here should be just one result --%>
          <c:forEach var="row" items="${idQuery.rows}">
               <c:set var="photoid" value="${row.NEXTVAL}"/>
          </c:forEach>

          <%-- Upload the media data into the database --%>
          <ord:storeMedia
             conn = "${myDS.connection}"
             table = "Photos"
             key = "${photoid}"
             keyColumn = "Id"
             mediaColumns = "Image"
             mediaParameters = "photograph"
             otherColumns = "Description, Location"
             otherValues = "<%=otherValuesVector%>"
          />

          <%-- update the thumbnail image column --%>
          <sql:update sql="call generateThumbnail(?)">
            <sql:param  value = "${photoid}"/>
          </sql:update>
```

```
        </sql:transaction>
      </c:otherwise>
    </c:choose>
  </ord:uploadFile>

</ord:uploadFormData>


<%-- HTML header --%>
<HTML LANG="EN">
<HEAD>
<TITLE>Oracle Multimedia JSP Tag Library Photograph Album Demonstration</TITLE>
</HEAD>

<%-- Direct the browser to the main page --%>
<META HTTP-EQUIV="REFRESH" CONTENT="2;URL=PhotoAlbum.jsp">

<BODY>

<%-- Page heading --%>
<TABLE BORDER="0" WIDTH="100%">
  <TR>
    <TD COLSPAN="2" BGCOLOR="#F7F7E7" ALIGN="CENTER"><FONT SIZE="+2">
      Oracle Multimedia JSP Tag Library Photograph Album Demonstration</FONT>
    </TD>
  </TR>
</TABLE>

<%-- Display header and instructions --%>
<P>
<FONT SIZE=3 COLOR="#336699">
<B>Photograph successfully uploaded into photograph album</B>
</FONT>
<HR SIZE=1>
</P>
<P>
Please click the link that follows or wait for the browser to refresh the page.
</P>

<%-- Output link to return to the main page --%>
<P>
<TABLE WIDTH="100%">
  <TR BGCOLOR="#F7F7E7">
    <TD COLSPAN="3" ALIGN="CENTER">
      <A HREF="PhotoAlbum.jsp">Return to photograph album</A>
    </TD>
  </TR>
</TABLE>
</P>

<%-- Finish the page --%>
</BODY>
</HTML>
```

The bolded Java, SQL, and HTML statements, in order of appearance in the `PhotoAlbumInsertPhoto.jsp` file, perform the following operations:

- Provide the required prefix and uri attributes in the JSP directives. The value of the prefix attribute specifies the XML namespace identifier, which must be

inserted before each occurrence of the library's tags in the JSP page. The value of the uri attribute indicates the location of the tag library descriptor (TLD) file for the specified tag library. In this example, the value `"ord"` specifies the XML namespace identifier for the Oracle Multimedia JSP Tag Library. And, the prefix attribute values `"sql"` and `"c"` specify the namespace identifiers for the JSTL sql and core tag libraries, respectively.

- Use the JSP tag uploadFormData to create a script variable named `fd`, which is an instance of the oracle.ord.im.OrdHttpUploadFormData object that holds the uploaded media.
- Use the JSP tag uploadFile to create the script variables: `ffName`, `sfName`, and `fLength`, which contain the full file name, short file name, and file length of the uploaded media, respectively.
- Use JSTL core tags to perform error checking.
- Use the JSTL sql tag to create a database DataSource.
- Use the JSTL sql tag to query the next unique id for the `photos` table in the database.
- Use a JSTL core tag to retrieve the next unique id from the query result.
- Use the JSP tag storeMedia to upload the media data into the `image` column of the `photos` table, and the description and location information into the `description` and `location` columns of the `photos` table.
- Use the JSTL sql tag to call a PL/SQL procedure to populate the `thumb` column of the `photos` table from the uploaded `image` column.

3

Media Retrieval Reference

Oracle Multimedia JSP Tag Library provides media retrieval tags that facilitate generating complete HTML multimedia tags or creating multimedia retrieval URLs for inclusion in the customized use of an HTML multimedia tag.

Media retrieval for a multimedia application includes the following tasks:

- Generating HTML tags to retrieve or render media data. The generated HTML tags include URLs that retrieve media data.

  The following example excerpt is from the `PhotoAlbum.jsp` file. This example shows the JSP tag embedImage.

  ```
  <ord:embedImage dataSourceName="jdbc/OracleDS"
                  table = "photos"
                  column = "thumb"
                  key = "<%=id%>"
                  keyColumn = "id"
                  alt = "<%=escapeHtmlString(description)%>"
                  border="1" />
  ```

  This example generates the following HTML <IMG> tag, which includes the URL that retrieves the image in the database:

  ```
  <img src="OrdGetMediaServlet?dataSourceName=
    jdbc%2OracleDS&table=photos&column=thumb&key=1&keyColumn=
    id&timeStamp=111078352&ext=.jpg" height=50 width=50
    alt="picture1" border="1" />
  ```

  See Chapter 2 for a complete description of the sample JSP application that uses tags from the Oracle Multimedia JSP Tag Library to retrieve media data from the database and deliver it to a browser. The browser then displays the media in a simple photograph album application.

- Using a media delivery component that retrieves media data from the database and delivers it to a browser. In the previous example, `OrdGetMediaServlet` is the media delivery component in the URL of the generated HTML <IMG> tag.

For more general purposes, Oracle Multimedia JSP Tag Library can generate HTML tags, with some level of customization capability, for the following media types:

- Images rendered using the HTML <IMG> tag.
- Audio and video data rendered using the HTML <EMBED> and <OBJECT> tags, where the media data is delivered directly from the database to a browser plug-in or other media player.

Note: For audio and video media data stored in Oracle Database that is accessed and delivered by RealNetworks streaming servers, Oracle recommends using Oracle Multimedia Plug-in for RealNetworks Streaming Servers rather than the Oracle Multimedia JSP Tag Library, for better performance.

In addition, Oracle Multimedia JSP Tag Library is flexible enough to permit the construction of HTML tags that render media data according to the requirements of the application, while using the tag library to construct media retrieval URLs and deliver the media to the browser.

Oracle Multimedia JSP Tag Library contains the following information about the media retrieval tags that operate on Oracle Multimedia objects:

- General Format for Media Retrieval Tags on page 3-3
- Media Retrieval Tags on page 3-13
- Media Retrieval URL Format on page 3-23

General Format for Media Retrieval Tags

Format

```
<ord:tagName [ custom-retrieval-attributes ]
             [ database-connection-attributes ]
             [ media-access-attributes ]
             [ media-cache-control-attributes ]
             [ table-and-column-attributes ]
             [ media-render-attributes ]>
</ord:tagName>
```

Description

Provide the ability to generate complete HTML media tags, such as <IMG>, or the ability to create media retrieval URLs for inclusion in the customized use of an HTML media tag. The media retrieval tags include the following tags under the prefix ord: mediaUrl, embedImage, embedAudio, and embedVideo.

Media retrieval tags have a set of common attributes, and tag-specific attributes that render media. This section describes the following common attributes and their parameters in detail: custom-retrieval-attributes, database-connection-attributes, media-access-attributes, media-cache-control-attributes, and table-and-column-attributes. The media-render-attributes are tag-specific; therefore, they are described in the sections for each media retrieval tag.

Usage Notes

When using media retrieval tags, the following information must be specified:

- Database connection information through which media data is retrieved from the database.
- Table, column, and key information that describes where the media data is to be retrieved from the database.

Thus, either database-connection-attributes and table-and-column-attributes or media-access-attributes must be specified.

custom-retrieval-attributes

Format

custom-retrieval-attributes =
retrievalPath = "string | <%= jspExpression %>"

Description

Offer a mechanism to supply the name of an application-specific media retrieval component for applications that require a high level of control over the delivery of media data. This retrieval component becomes part of the generated URL.

Parameters

retrievalPath
The path to the customized media retrieval component. See Media Retrieval URL Format for information about the input parameters to the media retrieval component.

Usage Notes

None.

Examples

Use the retrievalPath attribute to introduce a customized version of the `OrdGetMediaJsp.jsp` file, which operates as a media delivery JSP page:

```
<ord:embedVideo { database-connection-attributes }
                [ media-cache-control-attributes ]
                  retrievalPath = "customGet.jsp"
                { table-and-column-attributes }
                { media-attributes } />
```

database-connection-attributes

Format

database-connection-attributes =
dataSourceName = "string | <%= jspExpression %>"

Description

Specify the database connection. To retrieve media data from the database and deliver it to the browser, the media delivery component must have access to a database connection. Oracle Multimedia JSP Tag Library supports the use of data sources to specify connection properties.

The JDBC connection obtained from a data source must be able to support access to database objects and binary large objects (BLOBs) so that media data can be retrieved from Oracle Multimedia objects in the database. Thus, a native Oracle data source is required.

There are several native Oracle data sources. However, only data sources that support a connection caching mechanism are practical in a production environment. Other data sources, such as those that create a new JDBC database connection for each request to get a connection, result in extremely poor performance. This is especially noticeable for pages that include several multimedia data items, such as a set of thumbnail images.

To use a data source in a JSP page, you must define the data source, its JNDI name, and its connection and pooling properties. In OC4J, you define these attributes in a <data-source> element in the `data-source.xml` file. See *Oracle Containers for J2EE Services Guide* for more information about data sources.

The specified database connection is used by the media delivery component to retrieve media data from the database.

Parameters

dataSourceName
The name of a JDBC data source that can be retrieved using a default initial Java Naming and Directory Interface (JNDI) context.

Usage Notes

If you do not specify media-access-attributes, then you must specify database-connection-attributes and table-and-column-attributes.

Examples

Use the dataSourceName attribute to specify the database connection:

```
<ord:embedVideo dataSourceName = "empDB"
              [ media-cache-control-attributes ]
              [ custom-retrieval-attributes ]
              { table-and-column-attributes }
              { media-attributes } />
```

media-access-attributes

Format

```
media-access-attributes =
                    mediaAccessUnit = "string | <%= jspExpression %>"
                  [ key = "string | <%= jspExpression %>" ]
               [ rowid = "string | <%= jspExpression %>" ]
```

Description

Simplify the use of the retrieval tags. The information in the following attributes can be specified in the predefined `OrdJspTag.xml` file: database-connection-attributes, table-and-column-attributes, media-cache-control-attributes, and custom-retrieval-attributes. Then, in the retrieval tags, these attributes can be replaced with media-access-attributes, in which the mediaAccessUnit attribute is used to refer to the information defined in the `OrdJspTag.xml` file. If the mediaAccessUnit attribute and other attributes (for example: media-cache-control-attributes) are used together, the attribute defined in the tag overrides the attribute defined in the `OrdJspTag.xml` file. (See Appendix A for an example of this tag library configuration file.)

Parameters

mediaAccessUnit
A String literal or expression that specifies the name of the access-unit attribute defined in the `OrdJspTag.xml` file.

key
Row information that works with the mediaAccessUnit attribute to locate the media data in the table.

rowid
Row information that works with the mediaAccessUnit attribute to locate the media data in the table.

Usage Notes

If you do not specify database-connection-attributes and table-and-column-attributes, then you must specify media-access-attributes.

Either the key attribute or the rowid attribute must be specified in the tag.

Examples

Shows the mediaAccessUnit attribute being used in two ways, first with the key attribute and then with the rowid attribute. The first six lines of code represent a section of the `OrdJspTag.xml` file in which the access-unit attribute defines photoUnit.

```
<access-unit name="photoUnit"
                dataSourceName="myMediaDataDS"
                table="public_photos"
                keyColumn="ename"
                column="photo"
                     expiration="3600+60"/>

<ord:embedImage mediaAccessUnit="photoUnit">
```

```
                key = "SMITH"
              { media-render-attributes }/>
```

or:

```
<ord:embedImage mediaAccessUnit="photoUnit">
                rowid = "<%= empBean.getRowid() %>"
              { media-render-attributes }/>
```

media-cache-control-attributes

Format

```
media-cache-control-attributes =
                [ expiration = "string | <%= jspExpression %>" ]
            [ cache = "yes | no | no-remote | <%= jspExpression %>" ]
```

Description

Helps generate the Surrogate-Control response header for Web cache control. This response header enables the original Web server to dictate how surrogates are to handle response entities. The value of the expiration attribute in the tag is translated into the Surrogate-Control control max-age directive. The value of the cache attribute in the tag is translated into the Surrogate-Control no-store or no-store-remote directive.

Parameters

expiration

The value in the same format as the Surrogate-Control header's max-age directive is:

`expiration_time[+removal_time]`

where:

`expiration_time`: the amount of time during which the media object can be considered fresh, in seconds. After this time, the cache implementation must consider the cached object stale.

`removal_time`: the delay before the object is removed from the cache after the time expires.

If the expiration attribute is not specified, the max-age directive in the Surrogate-Control response header is set as infinity.

cache

The following table shows the valid values for this parameter:

Value	Description
`no`	The no-store directive in the Surrogate-Control response header
`no-remote`	The no-store-remote directive in the Surrogate-Control response header
`yes`	The Web cache is used

For more information about Oracle Web Cache, see *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache* in the Oracle Fusion Middleware Online Documentation Library.

Usage Notes

None.

Examples

Example 1: Shows how to use the expiration attribute.

```
<ord:embedVideo { database-connection-attributes }
                  expiration = "600"
                [ custom-retrieval-attributes ]
                { table-and-column-attributes }
                { media-attributes } />
```

Example 2: Shows how to use the cache attribute.

```
<ord:embedVideo { database-connection-attributes }
                  cache = "no"
                [ custom-retrieval-attributes ]
                { table-and-column-attributes }
                { media-attributes } />
```

table-and-column-attributes

Format

```
table-and-column-attributes =
                    table = "string | <%= jspExpression %>"
                    column = "string | <%= jspExpression %>"
                  [ key = "string | <%= jspExpression %>" ]
                  [ keyColumn = "string | <%= jspExpression %>" ]
                [ rowid = "string | <%= jspExpression %>" ]
```

Description

Specify where the media data is located in the database. There are three ways to identify the media data in the database:

- Use a primary key by specifying the key value with the key attribute.
- Use any column by specifying the column name with the keyColumn attribute and the column value with the key attribute.
- Use a ROWID by specifying the rowid attribute.

Parameters

table
A String literal or expression that specifies the name of the table containing the media data.

column
A String literal or expression that specifies the name of the column containing the media data.

key
A String literal or expression that specifies the key value to use to fetch the media. The table's primary key is used if the keyColumn attribute is not specified. If the table does not have a primary key, a key column name must be specified with the keyColumn attribute.

keyColumn
A String literal or expression that specifies the column to use to access the media.

rowid
A String literal or expression that indicates the ROWID of the media in the specified table. Specifying this attribute is the equivalent of specifying `key="rowid-value" keyColumn="rowid"`.

Usage Notes

If you do not specify media-access-attributes, then you must specify table-and-column-attributes and database-connection-attributes.

The attributes key, keyColumn, and rowid must be specified in one of the following combinations:

- key
- key and keyColumn
- rowid

Examples

Example 1: Shows how to use the table, column, and key attributes.

```
<ord:embedVideo { database-connection-attributes }
                [ media-cache-control-attributes ]
                [ custom-retrieval-attributes ]
                  table = "emp" column = "photo"
                  key = "<%= empBean.getEmpno() %>"
                { media-attributes } />
```

Example 2: Shows how to use the table, column, key, and keyColumn attributes.

```
<ord:embedVideo { database-connection-attributes }
                [ media-cache-control-attributes ]
                [ custom-retrieval-attributes ]
                  table = "emp" column = "photo"
                  key = "SMITH" keyColumn = "ename"
                { media-attributes } />
```

Example 3: Shows how to use the table, column, and rowid attributes.

```
<ord:embedVideo { database-connection-attributes }
                [ media-cache-control-attributes ]
                [ custom-retrieval-attributes ]
                  table = "emp" column = "photo"
                  rowid = "<%= empBean.getRowid() %>"
                { media-attributes } />
```

media-render-attributes

In the following sections, the media retrieval tags are defined with their specific media-render-attributes.

Oracle Multimedia supports image, audio, and video data. Thus, Oracle Multimedia JSP Tag Library retrieval tags support only those media types. No support is provided for other media types, such as text-based or application-specific types, that might be stored in the OrdDoc object type.

Media Retrieval Tags

This section presents reference information about these media retrieval tags, which operate on Oracle Multimedia objects:

embedAudio

Format

```
<ord:embedAudio [ database-connection-attributes ]
                [ table-and-column-attributes ]
                [ custom-retrieval-attributes ]
                [ media-access-attributes ]
                [ media-cache-control-attributes ]

                [ height = "number | <%= jspExpression %>" ]
                [ width = "number | <%= jspExpression %>" ]
                [ alt = "string | <%= jspExpression %>" ]
                [ helperApp = "mediaPlayer | realPlayer |
                            quicktimePlayer | <%= jspExpression %>" ]
                [ showControls = "true | false | <%= jspExpression %>" ]
                [ autoStart = "true | false | <%= jspExpression %>" ]
                [ loop = "true | false | <%= jspExpression %>" ]
                [ standby = "string | <%= jspExpression %>" ]
          [ audio = "<%= jspExpression %>" ] />
```

Description

Builds an HTML <OBJECT> tag and <EMBED> tag to embed an audio object in a page. This tag can specify the audio player to be used in the client's browser. It also defines a common set of audio attributes. See *Oracle Multimedia Reference* and *Oracle Multimedia User's Guide* for information about supported audio formats.

Parameters

height
The height of the displayed window, which overrides the default height of the displayed window.

width
The width of the displayed window, which overrides the default width of the displayed window.

alt
Brief alternate text that is displayed when the media cannot be retrieved or displayed.

helperApp
The name of the media player to be used by the browser to play the media. If a media player is not specified, the browser default player is activated. Currently, Oracle Multimedia JSP Tag Library supports three major media players: Windows Media Player, RealPlayer, and QuickTime Player. The Firefox browser, however, invokes a media player plug-in based on the MIME type of the media. Thus, the generated HTML tag has no control over the media player that is invoked by the Firefox browser.

showControls
Attribute that determines whether to show the control components of the player.

autoStart
Attribute that determines whether to play the audio automatically when it is retrieved.

loop
Attribute that determines whether to repeatedly play the audio.

standby
Attribute that specifies what to display when the audio is being retrieved.

audio
An instance of the oracle.ord.im.OrdAudio object. Because Oracle Multimedia JSP Tag Library must obtain the audio information from the oracle.ord.im.OrdAudio object, using the audio attribute is the most efficient way to use the Oracle Multimedia JSP Tag Library in cases where the JSP page has already fetched the row containing the audio object from the database.

Usage Notes

None.

Examples

See the examples in embedVideo.

embedImage

Format

```
<ord:embedImage [ database-connection-attributes ]
                [ table-and-column-attributes  [
                [ custom-retrieval-attributes ]
                [ media-access-attributes ]
                [ media-cache-control-attributes ]

                [ height = "number |  <%= jspExpression %>" ]
                [ width = "number | <%= jspExpression %>" ]
                [ border = "number | <%= jspExpression %>" ]
                [ align = "left | right | top | bottom | middle |
                          <%= jspExpression %>" ]
                [ alt = "string | <%= jspExpression %>" ]
                [ longdesc = "string | <%= jspExpression %>" ]
          [ image = "<%= jspExpression %>" ] />
```

Description

Builds an HTML <IMG> tag to embed an image in a page. A common set of <IMG> tag attributes, such as height, width, border, and align are also defined. The generated <IMG> tag always includes the height and width attributes. If these attributes are not specified in this tag, they are obtained either from the image object specified in this tag, or from the image object fetched from the database. See *Oracle Multimedia Reference* and *Oracle Multimedia User's Guide* for information about supported image formats.

Parameters

height
The height of the displayed window, which overrides the default height of the displayed window.

width
The width of the displayed window, which overrides the default width of the displayed window.

border
The border of the displayed image.

align
The alignment of the displayed image, which can be bottom, middle, top, left, or right.

alt
Brief alternate text that is displayed when the image cannot be retrieved or displayed.

longdesc
A link to the detailed (long) description of the image, which supplements the brief description provided by the alt attribute.

image
An instance of the oracle.ord.im.OrdImage object. Because Oracle Multimedia JSP Tag Library must obtain the image information from the oracle.ord.im.OrdImage object, using the image attribute is the most efficient way to use the Oracle Multimedia JSP

Tag Library in cases where the JSP page has already fetched the row containing the image object from the database.

Usage Notes

None.

Examples

Example 1: Shows how to use the border, alt, and align attributes in the embedImage tag.

```
<ord:embedImage dataSourceName = "empDB"
                table = "emp" column = "photo"
                key = "<%= empBean.getEmpno() %>"
                alt = "Employee photo"
                border = "1"
                align = "middle" />
```

The generated HTML tag would be as follows:

```
<img src="OrdGetMediaServlet?dataSourceName=
 empDB&table=emp&column=photo&key=1&timeStamp=111078352&ext=.jpg"
 height=256 width=256 alt="Employee photo" border=1 align="middle"/>
```

Example 2: Shows how to use the height and width attributes to override the actual height and width of the image.

```
<ord:embedImage dataSourceName = "empDB"
                table = "emp" column = "photo"
                key = "SMITH" keyColumn = "ename"
                height = "100" width = "100" />
```

The generated HTML tag would be as follows:

```
<img src="OrdGetMediaServlet?dataSourceName=
 empDB&table=emp&column=photo&key=1&keyColumn=
 "ename"&timeStamp=111078352&ext=.jpg"
 height=100 width=100/>
```

Example 3: Shows how to use the image attribute.

```
<ord:embedImage dataSourceName = "empDB"
                table = "emp" column = "photo"
                rowid = "<%= empBean.getRowId() %>"
                image = "<%= empBean.getPhoto() %>" />
```

The generated HTML tag would be as follows:

```
<img src="OrdGetMediaServlet?dataSourceName=
 empDB&table=emp&column=photo&rowid=AAAH2cAABAAAPAxAAA&
 timeStamp=111078352&ext=.jpg" height=256 width=256/>
```

where:

- empBean: is a JavaBean used to access an employee schema in a database. It includes get( ) methods that return column values, such as empno as a String object and photo as an OrdImage object, and database connection information.

embedVideo

Format

```
<ord:embedVideo [ table-and-column-attributes ]
                [ database-connection-attributes ]
                [ custom-retrieval-attributes ]
                [ media-access-attributes ]
                [ media-cache-control-attributes ]

                [ height = "number | <%= jspExpression %>" ]
                [ width = "number | <%= jspExpression %>" ]
                [ alt = "string | <%= jspExpression %>" ]
                [ helperApp = "mediaPlayer | realPlayer |
                        quicktimePlayer | <%= jspExpression %>" ]
                [ showControls = "true | false | <%= jspExpression %>" ]
                [ autoStart = "true | false | <%= jspExpression %>" ]
                [ loop = "true | false | <%= jspExpression %>" ]
                [ standby = "string | <%= jspExpression %>" ]
          [ video = "<%= jspExpression %>" ] />
```

Description

Builds an HTML <OBJECT> tag and <EMBED> tag to embed a video object in a page. This tag can specify the video player to be used in the client's browser. It also defines a common set of video attributes. See *Oracle Multimedia Reference* and *Oracle Multimedia User's Guide* for information about supported video formats.

Parameters

height
The height of the displayed window, which overrides the default height of the displayed window.

width
The width of the displayed window, which overrides the default width of the displayed window.

alt
Brief alternate text that is displayed when the media cannot be retrieved or displayed.

helperApp
The name of the media player to be used by the browser to play the media. If a media player is not specified, the browser default player is activated. Currently, Oracle Multimedia JSP Tag Library supports three major media players: Windows Media Player, RealPlayer, and QuickTime Player. The Firefox browser, however, invokes a media player plug-in based on the MIME type of the media. Thus, the generated HTML tag has no control over the media player that is invoked by the Firefox browser.

showControls
Attribute that determines whether to show the controls components of the player.

autoStart
Attribute that determines whether to play the video automatically when it is retrieved.

loop
Attribute that determines whether to repeatedly play the video.

standby
Attribute that specifies what to display when the video is being retrieved.

video
An instance of the oracle.ord.im.OrdVideo object. Because Oracle Multimedia JSP Tag Library must obtain the video information from the oracle.ord.im.OrdVideo object, using the video attribute is the most efficient way to use the Oracle Multimedia JSP Tag Library in cases where the JSP page has already fetched the row containing the video object from the database.

Usage Notes

None.

Examples

This example shows how to use the embedVideo tag with its media-render-attributes. In the example, the helperApp attribute includes all the possible choices of media players.

The following example uses the same JavaBean (empBean) as in the example for embedImage. The three samples that follow show HTML output for the three media players.

```
<ord:embedVideo dataSourceName = "empDB"
                table = "emp" column = "greetings"
                key = "<%= empBean.getEmpno() %>"
                alt = "Employee greetings"
                height = 240 width = 300
                helperApp = "mediaPlayer|realPlayer|quicktimePlayer"
                showControls = "true"
                autoStart = "true"
                loop = "true"
                  standby = "Loading media player components ..." />
```

Sample Output 1: Shows the generated HTML tag for Windows Media Player. It uses video data with the file extension .avi, and with the value of the helperApp attribute set to mediaPlayer.

```
<OBJECT
   ID="mediaPlayer"
   CLASSID= "clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
   CODEBASE= "http://activex.microsoft.com/activex/controls/
     mplayer/en/nsmp2inf.cab#Version=5,1,52,701"
   TYPE="application/x-oleobject"
   STANDBY="Loading media player components..."
   HEIGHT="240" WIDTH="300">
   <PARAM NAME="showcontrols" VALUE="true">
   <PARAM NAME="autostart" VALUE="true">
   <PARAM NAME="loop" VALUE="true">
   <PARAM NAME="filename"   VALUE="OrdGetMediaServlet?dataSourceName=empDB&table=
     emp&column=greetings&key=1&timeStamp=111078352&ext=.avi">

   <EMBED TYPE="video/x-msvideo"
          STANDBY="Loading media player components…"
          CONTROLLER="true"
          CONTROLS="ImageWindow,ControlPanel"
```

```
          SHOWCONTROLS="true"
          AUTOSTART="true"
          LOOP="true"
          HEIGHT="240"  WIDTH="300"

 SRC="OrdGetMediaServlet?dataSourceName=empDB&table=
  emp&column=greetings&key=1&timeStamp=111078352&ext=.avi" >
```

where:

- .avi: is the file extension for Microsoft AVI media.
- mediaPlayer: is the value of the helperApp attribute.
- video/x-msvideo: is the MIME type of the media data.

Sample Output 2: Shows the generated HTML tag for RealPlayer. It uses video data with the file extension .rm, and with the value of the helperApp attribute set to realPlayer.

```
<OBJECT
   ID="RVOCX"
   CLASSID= "clsid: CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA"
   STANDBY="Loading media player components..."
   HEIGHT="240" WIDTH="300" >
   <PARAM NAME="controls" VALUE="ImageWindow, ControlPanel">
   <PARAM NAME="autostart" VALUE="true">
   <PARAM NAME="loop" VALUE="true">
   <PARAM NAME="filename" VALUE="OrdGetMediaServlet?dataSourceName=empDB&table=
     emp&column=greetings&key=1&timeStamp=111078352&ext=.rm">

   <EMBED TYPE="audio/x-pn-realaudio-plugin"
          STANDBY="Loading media player components…"
          CONTROLLER="true"
          CONTROLS="ImageWindow,ControlPanel"
          SHOWCONTROLS="true"
          AUTOSTART="true"
          LOOP="true"
          HEIGHT="240"  WIDTH="300"
 SRC="OrdGetMediaServlet?dataSourceName=empDB&table=
  emp&column=greetings&key=1&timeStamp=111078352&ext=.rm" >
   <NOEMBED>
   <P>Employee Greetings.</P>
   </NOEMBED>
</OBJECT>
```

where:

- .rm: is the file extension for RealNetworks Real Video media.
- realPlayer: is the value of the helperApp attribute.
- audio/x-pn-realaudio-plugin: is the MIME type of the media data.

Sample Output 3: Shows the generated HTML tag for QuickTime Player. It uses video data with the file extension .mov, and with the value of the helperApp attribute set to quicktimePlayer.

```
<OBJECT
   CLASSID= "clsid: 02BF25D5-8C17-4B23-BC80-D3488ABDDC6B"
```

```
   CODEBASE= "http://www.apple.com/qtactivex/qtplugin.cab"
   STANDBY="Loading media player components..."
   HEIGHT="240" WIDTH="300" >
   <PARAM NAME="controller" VALUE="true">
   <PARAM NAME="autostart" VALUE="true">
   <PARAM NAME="loop" VALUE="true">
   <PARAM NAME="filename"   VALUE="OrdGetMediaServlet?dataSourceName=empDB&table=
     emp&column=greetings&key=1&timeStamp=111078352&ext=.mov">

   <EMBED TYPE="video/quicktime"
          STANDBY="Loading media player components…"
          CONTROLLER="true"
          CONTROLS="ImageWindow,ControlPanel"
          SHOWCONTROLS="true"
          AUTOSTART="true"
          LOOP="true"
          HEIGHT="240"  WIDTH="300"
 SRC="OrdGetMediaServlet?dataSourceName=empDB&table=
  emp&column=greetings&key=1&timeStamp=111078352&ext=.mov" >
   <NOEMBED>
   <P>Employee Greetings.</P>
   </NOEMBED>
</OBJECT>
```

where:

- .mov: is the file extension for Apple QuickTime media.
- quicktimePlayer: is the value of the helperApp attribute.
- video/quicktime: is the MIME type of the media data.

mediaUrl

Format

```
<ord:mediaUrl [ database-connection-attributes ]
              [ table-and-column-attributes ]
              [ custom-retrieval-attributes ]
              [ media-access-attributes ]
              [ media-cache-control-attributes ]

               id = "string" >

                 tag body...

</ord:mediaUrl>
```

Description

Generates a media retrieval URL object that can be used in the tag body.

Parameters

id
The name of the script variable. The URL of the media objects in the database can be obtained by calling the getUrl( ) method while using this script variable.

Usage Notes

None.

Examples

Example 1: Use the generated URL in the customized HTML <IMG> tag:

```
<ord:mediaUrl dataSourceName = "empDB"
              table = "emp" column = "photo"
              key = "1"
              id= "photo" >

     <img src="<%= photo.getUrl( ) %>" onmouseover="..." ...>

</ord:mediaUrl>
```

Example 2: Use the generated URL within the HTML <A> (link) tag to locate an image file in an employee database:

```
<ord:mediaUrl dataSourceName = "empDB"
              table = "emp" column = "photo"
              key = "1"
              id = "photo" >

   <a href="<%= photo.getUrl() %>"> Click here to view image </a>

</ord: mediaUrl>
```

Media Retrieval URL Format

Format

```
<mediaRetrievalPath>?
 <databaseConnection>&
 <tableInfo>&
 <columnInfo>&
 <rowInfo>&
 <expiration>&
 <cache>&
 <updateTimeStamp>&
 <fileExtension>
```

Description

Specifies the format of the URL string generated by the media retrieval tags. This URL string is passed to the media delivery component to retrieve the media data.

Parameters

mediaRetrievalPath
The value of the retrievalPath attribute specified in the media retrieval tag. By default, the value is OrdGetMediaServlet.

databaseConnection
dataSourceName=name where:

name: is the Data Source name specified in the tag.

tableInfo
table=tableName where:

tableName: is specified in the tag.

columnInfo
column=columnName where:

columnName: is specified in the tag.

rowInfo
One of the following, depending on the row information specified in the tag:

- key=keyVal
- rowid=rowidVal
- key=keyVal&keyColumn=keyColumnName

expiration
The optional cache control attribute expiration=expirationValue where:

expirationValue: is specified in the tag.

cache
The optional cache control attribute cache=cacheValue where:

cacheValue: is specified in the tag.

updateTimeStamp
timeStamp=timestampvalue where:

timestampvalue: is the last update time of the media object.

fileExtension
ext=.fileExt where:

fileExt: is the file extension of the media.

Usage Notes

The SRC attribute in the HTML <IMG>, <OBJECT>, and <EMBED> tags, which are generated by the Oracle Multimedia JSP Tag Library tags embedImage, embedAudio, and embedVideo respectively, also follows this format. For more information about these JSP tags, see embedImage, embedAudio, and embedVideo.

Examples

See the examples in embedImage.

4

Media Upload Reference

Oracle Multimedia JSP Tag Library provides media upload tags that help multimedia applications to upload media data into the database.

Media upload tags meet the following requirements:

- Make it easier to handle multipart form-data requests.
- Simplify the storage of media data in the database.

Application developers must be able to specify the following information when attempting to upload media data:

- Database connection information through which media data is written to the database
- Table, column, and key information that describes where the media data is to be uploaded in the database

See Chapter 2 for a complete description of the sample JSP application that uses tags from the Oracle Multimedia JSP Tag Library to upload media files into a database.

Oracle Multimedia JSP Tag Library contains the following information about the media upload tags that operate on Oracle Multimedia objects:

Media Upload Tags

This section presents reference information about these media upload tags, which operate on Oracle Multimedia objects:

storeMedia

Format

```
<ord:storeMedia   conn = "<%= jspExpression %>"
                  table = "string | <%= jspExpression %>"
                  [ key = "string | <%= jspExpression %>" ]
                  [ keyColumn = "string | <%= jspExpression %>" ]
                  [ rowid = "string | <%= jspExpression %>" ]
                  mediaColumns = "values"
                  mediaParameters = "values"
                  [ otherColumns = "values" ]
                  [ otherValues = "<%= jspExpression %>" ]/>
```

where,

values = string | <%=jspExpression%> [, string | <%=jspExpression%>]...

Description

Operates within the body of the uploadFormData tag. The storeMedia tag implicitly uses the form-data object created by the uploadFormData tag through the mediaParameters attribute. This tag loads the uploaded media data from the HTML form into the OrdImage, OrdAudio, OrdVideo, or OrdDoc object in the specified table, row, and column in the database. If the specified row does not exist, a new row is inserted and updated with the loaded data. The transaction commits automatically if autocommit is set to true in the database connection object.

The mediaColumns and mediaParameters attributes enable the uploading of multiple media data specified in an HTML form into multiple columns in a particular row. The otherColumns and otherValues attributes enable the updating of other columns of the table along with the media data.

After loading the media data in the database BLOB, the tag calls the setProperties( ) method to set the media properties within the object. If the media format is not recognized by Oracle Multimedia, this tag sets the mimeType and length properties only.

> **Note:** In the case when the specified row does not exist and a new row is inserted into the table, any other columns in the table that are not specified in the mediaColumns and otherColumns attributes must permit the null value.

Parameters

conn
The JDBC connection (java.sql.Connection object) used to store the media data.

table
A String literal or expression that specifies the name of the table containing the media data.

key
A String literal or expression that specifies the key value to use to fetch the media. The table's primary key is used if the keyColumn attribute is not specified. If the table does

not have a primary key, a key column name must be specified with the keyColumn attribute.

keyColumn
A String literal or expression that specifies the column to use to access the media.

rowid
A String literal or expression that indicates the ROWID of the media in the specified table. Specifying this attribute is the equivalent of specifying `key="rowid-value" keyColumn="rowid"`.

mediaColumns
The names of the columns to be loaded with the uploaded media data.

mediaParameters
The names of the parameters used to upload the media data, as specified in the HTML form.

otherColumns
The names of the other columns to be updated or inserted in the table.

otherValues
The updated or inserted values of the other columns in the table that correspond to the otherColumns attribute. These values must be stored in a java.util.Vector object.

Exceptions

None.

Usage Notes

None.

Examples

Example 1: Shows how to upload an image to a particular row in the table, and update other columns in that row at the same time.

```
<ord:uploadFormData formDataId="fd">
 <%
     java.util.Vector otherValuesVector = new java.util.Vector();
     otherValuesVector.add(fd.getParameter("desc"));
     otherValuesVector.add(fd.getParameter("loc"));
   %>
    <ord:storeMedia conn = "<% albumBean.getConnection() %>"
         table = "photos"
         key = "<%= fd.getParameter( "desc" ) %>"
         mediaColumns = "photo"
         mediaParameters = "photo"
         otherColumns = "Description, Location"
         otherValues = "<%= otherValuesVector %>" />
</ord:uploadFormData>
```

Example 2: Shows how to upload image and audio data into a particular row in a table, at the same time.

```
<ord:uploadFormData formDataId="fd">
    <%
     java.util.Vector otherValuesVector = new java.util.Vector();
     otherValuesVector.add(fd.getParameter("desc"));
```

```
     %>
    <ord:storeMedia conn = "<% albumBean.getConnection() %>"
         table = "photos"
         key = "<%= fd.getParameter( "desc" ) %>"
         mediaColumns = "photo, music"
         mediaParameters = "photo, music"
         otherColumns = "Description"
         otherValues = "<%= otherValuesVector %>" />
</ord:uploadFormData>
```

uploadFile

Format

```
<ord:uploadFile   parameter = "string | <%= jspExpression %>"
                  [ mimeType = "string" ]
                  [ length = "string" ]
                  [ fullFileName = "string" ]
                  [ shortFileName = "string" ]
                  [ inputStream = "string" ] >
    file-information-processing
</ord:uploadFile>
```

Description

Provides access to uploaded file information. This tag uses the object created by the uploadFormData tag implicitly, through the parameter attribute. This tag must be nested within the uploadFormData tag.

Parameters

parameter
The name of the parameter used to upload the file, as specified in the HTML form.

mimeType
The script variable name for the MIME type of the file.

length
The script variable name for the length of the file.

fullFileName
The script variable name for the file's full file name.

shortFileName
The script variable name for the file's short file name.

inputStream
The script variable name for the java.io.InputStream object for the uploaded file.

Exceptions

None.

Usage Notes

None.

Examples

Use an uploadFile tag nested within an uploadFormData tag to enable access to various file attributes, including the MIME type, length, and name:

```
<ord:uploadFormData formDataId="fd">
  <%
    String desc = fd.getParameter( "desc" );
    String loc fd.getParameter( "loc" );
    <ord:uploadFile parameter = "photo"
```

```
                    mimeType = "photoType"
                    length = "photoLength"
                    fullFileName = "photoFullName"
                    shortFileName = "photoShortName">
                    inputStream = "photoIn">

   The mimeType of the file is <%=photoType%>
   The length is <%=photoLength%>
                .
                .
                .

   </ord:uploadFile>
  %>
</ord:uploadFormData>
```

where:

- photo: is the parameter name of the uploaded file in the HTML form.
- photoType: is the name of the variable that contains the MIME type of the media.
- photoLength: is the name of the variable that contains the file length of the uploaded file.
- photoFullName: is the name of the variable that contains the full file name of the uploaded file.
- photoShortName: is the name of the variable that contains the short file name of the uploaded file.
- photoIn: is an instance of the java.io.InputStream object for the uploaded file.

uploadFormData

Format

```
<ord:uploadFormData   formDataId = "string"
                [ releaseFormData = "true | false | <%= jspExpression %>" ]
                [ maxMemory = "number | <%= jspExpression %>" ]
                [ tempDir = "string | <%= jspExpression %>" ]
     form-data-processing
</ord:uploadFormData>
```

Description

Parses the multipart/form-data HTTP request to provide access to text-based form parameters and the contents of uploaded files transmitted from a browser to a Web server.

Parameters

formDataId
The name of the script variable of type oracle.ord.im.OrdHttpUploadFormData from which the JSP page can obtain text-based form parameters and the contents of the uploaded files.

releaseFormData
Attribute used to retain uploaded media for processing after the end of the uploadFormData tag, such as in an included JSP page or a JSP page to which the request is forwarded. By default, the value of this attribute is true, which releases all uploaded media at the end of the tag. To retain the uploaded media, set the value of this attribute to false. Then, call the release( ) method in the OrdHttpUploadFormData object to release the resource when the uploaded media is no longer needed.

maxMemory
Attribute that specifies the maximum amount of memory that can be consumed by the uploaded media for one HTTP POST request before storing the media temporarily on disk. By default, all uploaded media is stored in memory until it is loaded into the database by the application and released at the end of the tag. In cases where users may want to upload large media files, such as large video clips, application developers can choose to store uploaded files temporarily on disk, before they are loaded into the database and released.

tempDir
Attribute that lets you override the default behavior. If this attribute is not specified, by default, uploaded media that is stored on disk temporarily is stored in one of the two locations identified by the following properties: javax.servlet.context.tempdir and java.io.tmpdir

where:

- javax.servlet.context.tempdir, if present, takes precedence over java.io.tmpdir.

> **Note:** If execution of the JSP page is interrupted before the end of the uploadFormData tag, any temporary files are deleted either during garbage collection or when the user's session ends, whichever occurs sooner.

Exceptions

None.

Usage Notes

This tag creates the form-data object, which is an object of type oracle.ord.im.OrdHttpUploadFormData. Access to upload form data is provided by the following methods:

- String getParameter(String name)
- String [ ] getParameterValues(String name)
- Enumeration getParameterNames( )
- OrdHttpUploadFile getFileParameter(String name)
- OrdHttpUploadFile [ ] getFileParameterValues(String name)
- Enumeration getFileParameterNames( )

Examples

Use the tag uploadFormData to create a form-data object, which the JavaBean called albumBean uses to access the text-based form parameters and the uploaded file content:

```
<ord:uploadFormData formDataId="fd">
  <%
    albumBean.setDescription( fd.getParameter( "desc" ) );
    albumBean.setLocation( fd.getParameter( "loc" ) );
    albumBean.insertNewEntry();
    albumBean.fetchRow();
    albumBean.loadPhoto( fd.getFileParameter( "photo" ) );
    albumBean.updateRow();
    albumBean.commit();
  %>
</ord:uploadFormData>
```

A

Configuring Oracle Multimedia JSP Tag Library

This appendix provides information on configuring the Oracle Multimedia JSP Tag Library within Oracle Database and other environments.

This appendix includes the following sections:

For details about installation and configuration, see the online `README.txt` file included in this kit.

> **Note:** In this appendix, *<ORACLE_HOME>* represents the Oracle home directory.

A.1 Prerequisite Products

You must have installed the following products on your system before using the Oracle Multimedia JSP Tag Library:

- Oracle Database 11*g*

  For information about Oracle Database, visit the Oracle Technology Network Web site at

  `http://otn.oracle.com/index.html`

- Oracle Fusion Middleware 11*g*

  For information about Oracle Fusion Middleware, visit the Oracle Technology Network Web site at

  `http://otn.oracle.com/software/products/ias/index.html`

A.2 Oracle Multimedia JSP Tag Library Components

The Oracle Multimedia JSP Tag Library is packaged with a set of components in the JAR file `ordjsptag.jar`. Table A–1 lists the components in this file.

Table A–1 Components of the Oracle Multimedia JSP Tag Library

Name	Description
`ordim-tag-library`	The Java class library
`multimedia-taglib.tld`	The tag library descriptor (TLD) file
`OrdGetMediaServlet`	The default media delivery component

The Oracle Multimedia JSP Tag Library JAR file `ordjsptag.jar` depends on the following Oracle Multimedia library JAR files:

- `ordim.jar` - Contains the Oracle Multimedia Java library.
- `ordhttp.jar` - Contains the Oracle Multimedia Servlets and JSP Java library.

Thus, to use the Oracle Multimedia JSP Tag Library, applications must package the JAR files `ordim.jar` and `ordhttp.jar` with the application or configure those files with the application server.

The Oracle Multimedia library JAR files are available with the Oracle installation, in the directory *<ORACLE_HOME>*`/ord/jlib`.

A.3 Configuring Oracle Multimedia JSP Tag Library with Applications

This section describes configuration parameters and deployment configurations that you might have to specify when deploying your application with the Oracle Multimedia JSP Tag Library.

A.3.1 Specifying the TLD File in a JSP Page

The Oracle Multimedia JSP Tag Library tag library descriptor (TLD) file is included in the Oracle Multimedia JSP Tag Library JAR file (`ordjsptag.jar`). To use the Oracle Multimedia JSP Tag Library in a Web application JSP page, Web applications must use the following namespace as the value of the uri attribute for the JSP taglib directive:

`http://xmlns.oracle.com/jsp/ord/multimedia-taglib.tld`

A.3.2 Specifying the Media Delivery Component

Oracle Multimedia JSP Tag Library provides two media delivery components, a media delivery servlet and a media delivery JSP page. You can specify the media delivery JSP page or use the media delivery servlet by default. Web applications can specify which media delivery component to use by defining the media-retrieval-path element in the tag library configuration file `OrdJspTag.xml`.

The media delivery servlet, oracle.ord.im.jsp.OrdGetMediaServlet, is packaged with the tag library. It is the default media delivery component. Specify the virtual path to the servlet in the `web.xml` file for your Web application, similar to Example A–1:

Example A–1 Virtual Path Specification for the Media Delivery Servlet

```
<servlet>
    <servlet-name>oracle_ord_im_jsp_media_servlet</servlet-name>
    <servlet-class>oracle.ord.im.jsp.OrdGetMediaServlet</servlet-class>
</servlet>

  <servlet-mapping>
    <servlet-name>oracle_ord_im_jsp_media_servlet</servlet-name>
    <url-pattern>/OrdGetMediaServlet</url-pattern>
```

```
</servlet-mapping>
```

You can download the media delivery JSP page, `OrdGetMediaJsp.jsp`, from OTN. First, define the JSP page in the tag library configuration file. Then, copy it to the JSP directory for your Web application.

Using a JSP page to deliver media data is convenient because it requires no additional configuration beyond copying the supplied page into the JSP page directory for the application. For performance reasons, however, it might be desirable, or even necessary, to deliver media data from a servlet, rather than a JSP page.

A.3.3 Authorizing Access to Media Data

All Oracle Multimedia JSP Tag Library actions that construct URLs also register table name and column name information within each user session. By default, the media delivery components verify this information when responding to a media retrieval request. If the verification check fails, media retrieval requests are rejected. Such verification ensures that malicious users cannot retrieve media data from any data source, table, or column using user-created URLs.

Some applications must support general access to specific tables and columns. For example, applications where media URLs might be exchanged by users in mail messages might need such access to data. For these applications, a mechanism is provided to enable anyone to access columns in tables in specific data sources. This mechanism involves setting the access-control attribute within the tag library configuration file `OrdJspTag.xml`.

Note: To use this mechanism, applications must specify database connection information using a dataSourceName attribute in the tag.

A.3.4 Setting Cache Control Attributes

Oracle Multimedia JSP Tag Library permits limited setting of cache control attributes. In this version, you can set the max-age, no-store, or no-store-remote directives of the Surrogate-Control header for all media retrieved from a particular named column in a specified table. The values of these directives are specified by the tag attributes expiration and cache.

expiration Attribute

The format for the expiration attribute is as follows:

```
expiration_time[+removal_time]
```

where:

- expiration_time: the time-to-live (TTL), in seconds.
- removal_time: the delay, in seconds, before the resource is removed from the cache after it expires.

This example specifies the expiration time only: `1800`

This example specifies both the expiration time and the removal time: `10000+600`

cache Attribute

The values for the cache attribute are: `no`, `no-remote`, or `yes`.

The values `no` and `no-remote` are translated into the no-store and no-store-remote directives in the Surrogate-Control header, respectively.

The value `yes` means that the Web cache is used.

> **Note:** Applications that require more fine-grained control over setting cache control attributes must either avoid using the Oracle Multimedia JSP Tag Library or design their own media delivery mechanism by replacing the default media delivery component.

For more information about Oracle Web Cache, see media-cache-control-attributes. See also *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache* in the Oracle Fusion Middleware Online Documentation Library.

A.3.5 Sample Tag Library Configuration File

This section describes the sample tag library configuration file `OrdJspTag.xml`. Because this XML file is application-specific, place it in the same directory as the `web.xml` file for the Web application. The tag library configuration file is read at the first requirement, and then put into application context for further retrieval.

Example A–2 shows the sample tag library configuration file `OrdJspTag.xml`.

Example A–2 Sample Configuration File

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE ordim-tag-library [
<!ELEMENT ordim-tag-library (media-control?, media-access?, media-player*,
    media-retrieval-path?)>
<!ELEMENT media-control (data-source?)>
<!ELEMENT data-Source (table+)>
<!ELEMENT table (column+)>
<!ELEMENT column (access-control)>
<!ELEMENT access-control EMPTY>
<!ELEMENT media-access (access-unit+)>
<!ELEMENT access-unit EMPTY>
<!ELEMENT media-player (class-id)>
<!ELEMENT class-id EMPTY>
<!ELEMENT media-retrieval-path (#PCDATA)>
<!ATTLIST media-control filtering (true | false) #IMPLIED>
<!ATTLIST data-source name CDATA #REQUIRED>
<!ATTLIST table name CDATA #REQUIRED>
<!ATTLIST column name CDATA #REQUIRED>
<!ATTLIST access-control enabled (true | false) #REQUIRED>
<!ATTLIST access-unit name CDATA #REQUIRED>
<!ATTLIST access-unit dataSourceName CDATA #IMPLIED>
<!ATTLIST access-unit table CDATA #REQUIRED>
<!ATTLIST access-unit keyColumn CDATA #IMPLIED>
<!ATTLIST access-unit column CDATA #REQUIRED>
<!ATTLIST access-unit expiration CDATA #IMPLIED>
<!ATTLIST access-unit cache (yes|no|no-remote) #IMPLIED>
<!ATTLIST media-player name (mediaPlayer|realPlayer|quicktimePlayer) #REQUIRED>
<!ATTLIST class-id clsid CDATA #REQUIRED>
 ]>

<ordim-tag-library>
<media-control filtering="true">
```

```
  <data-source name="myMediaDataDS">
    <table name="public_photos">
      <column name="photo">
        <access-control enabled="false"/>
      </column>
      <column name="thumb">
        <access-control enabled="false"/>
      </column>
    </table>
    <table name="public_videos">
      <column name="movie">
        <access-control enabled="false"/>
      </column>
    </table>
  </data-source>
</media-control>
<media-access>
  <access-unit name="photoUnit"
                dataSourceName="myMediaDataDS"
                table="public_photos"
                keyColumn="ename"
                column="photo"
                expiration="3600+60"/>
  <access-unit name="thumbUnit"
                dataSourceName="myMediaDataDS"
                table="public_photos"
                keyColumn="ename"
                column="photo"
                expiration="3600+60"/>
  <access-unit name="movieUnit"
                dataSourceName="myMediaDataDS"
                table="public_videos"
                column="movie"
                expiration="3600+60"
                cache="no"/>
  <access-unit name="mp3Unit"
                table="musics"
                column="mp3"
                expiration="3600+60"
                cache="no"/>
</media-access>
<media-player name="mediaPlayer">
  <class-id clsid="22D6F312-B0F6-11D0-94AB-0080C74C7E95"/>
</media-player>
<media-retrieval-path>
  GetMediaServlet
</media-retrieval-path>
</ordim-tag-library>
```

The `OrdJspTag.xml` file contains these elements, which provide the media retrieval tags with flexibility and simplicity:

- media-control
- media-access
- media-player
- media-retrieval-path

media-control Element

The media-control element is used to control access to the media data. This element is optional. By default, media data can be accessed only within the session in which the media retrieval tags are used. However, to make the media data accessible to all clients (including clients outside the session), the application manager can set the enabled attribute of the access-control subelement to `false`.

The filtering attribute of the media-control subelement can be set to `true`. This setting ensures that the media delivery component filters the special characters when HTML files are delivered to the browser.

media-access Element

The media-access element is used to define media access units, thus simplifying usage of the media retrieval tags. This element is optional. Each access-unit subelement defines a media access unit with a name and attributes that correspond to the following media retrieval tag common attributes: database-connection-attributes, table-and-column-attributes, and media-cache-control-attributes. The attributes defined in each media access unit can be overridden by the attributes defined in the media retrieval tag.

media-player Element

The media-player element defines the class identifier (clsid) used in the generated HTML <OBJECT> tag for the media player. This element is defined only when an application requires a specific media player in the Internet Explorer browser.

media-retrieval-path Element

The media-retrieval-path element is used to specify the media delivery component used by the Oracle Multimedia JSP Tag Library. This element is defined only when an application requires a media delivery component other than the default media delivery component (OrdGetMediaServlet).

B

Oracle Multimedia JSP Tag Library Error Messages

This appendix contains information on the errors that can be raised by the Oracle Multimedia JSP Tag Library. These errors may be thrown when trying to get the contents of a public object.

Not upload request.

Cause: Upload request was not encoded using the multipart/form-data encoding format.

Action: Use the multipart/form-data encoding format to send the upload request.

Tag <tag name> must be nested within uploadFormData tag.

Cause: The specified tag was not nested within the uploadFormData tag.

Action: The specified tag must appear in the body of the uploadFormData tag.

No file is specified in form field: <field name>.

Cause: The value of the parameter attribute in the uploadFile tag was incorrect.

Action: Match the value of the parameter attribute in the uploadFile tag to the correct name in the multipart/form-data upload request.

Inconsistent media data and media column(s).

Cause: There was no match between the media data types and the media column(s).

Action: Check the values of the mediaParameters and mediaColumns attributes.

Invalid player: <player name>.

Cause: The specified media player was invalid.

Action: Check the value of the helperApp attribute to make sure it specifies a valid media player.

Incorrect input parameter <parameter name> for the uploaded file.

Cause: The value of the mediaParameters attribute was incorrect.

Action: Check the value of the mediaParameters attribute.

Incorrect media column to retrieve <column name>.

Cause: The media column type did not match the type specified by the retrieval tag.

Action: Make sure the media column is correct.

Configuration file not found.

Cause: Unable to find the Oracle Multimedia JSP Tag Library configuration file.

Action: Check to make sure the Oracle Multimedia JSP Tag Library configuration file exists, and is in the same directory as the `web.xml` file for the Web application.

Access unit <access unit name> not found.

Cause: The specified access unit name was not defined.

Action: Define the access-unit attribute in the configuration file.

No database connection information specified.

Cause: The database connection information was not specified.

Action: Specify the connectionCache or dataSourceName parameter.

Unable to establish database connection.

Cause: No connection information, or incorrect connection information, was provided.

Action: Check the connection information to make sure it is correct.

Unable to get the keyColumn name.

Cause: Unable to get the column name of the primary key.

Action: Make sure the primary key exists or specify the column attribute.

Failed to retrieve Oracle Multimedia objects.

Cause: An error occurred while trying to retrieve the Oracle Multimedia object from the database.

Action: Check the database to make sure the object exists.

Failed to update Oracle Multimedia objects.

Cause: An error occurred while trying to update the Oracle Multimedia object from the database.

Action: Contact Oracle Support Services.

Column <column name> is not an Oracle Multimedia object type.

Cause: The column specified by the mediaColumns attribute was not a valid Oracle Multimedia object type.

Action: Check the value of the mediaColumns attribute.

Table or column name not provided.

Cause: No table name or media column name was provided.

Action: Specify the table name and column name.

Cannot locate the row.

Cause: Unable to locate the row in the table.

Action: Check the value of the key or rowid attribute to make sure it is correct.

Column names and column values are not matched.

Cause: There was no match between either the media column names and their values or the other column names and their values.

Action: Check the values of the following attributes: mediaParameters, mediaColumns, otherColumns, and otherValues.

Error reading the configuration file.

Cause: An I/O error occurred while trying to read the Oracle Multimedia JSP Tag Library configuration file.

Action: Check the Oracle Multimedia JSP Tag Library configuration file to make sure it is a valid XML file that conforms to the rules of the Document Type Definition (DTD) shown in the sample configuration file provided in Appendix A.

Index

M

O

P

R

S

T

U