Development-Excel Generation

For a "clean" copy of this file, head over to http://www.overhyped.com./downloads/OTCPOIArchive.java

```
/*
Copyright (c) 2003 Overhyped Technologies, LLC.

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

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

package com.overhyped.webobjects;

import com.webobjects.foundation.*;
import com.webobjects.appserver.*;
import com.webobjects.eocontrol.*;
import com.webobjects.eoaccess.*;
import com.webobjects.directtoweb.*;

import java.io.*;
import java.util.Date;

import org.apache.poi.hssf.usermodel.*;

public class OTCPOIArchive extends Object {
  protected D2WContext d2wContext;
  protected EOEnterpriseObject object;
  static String _fileUploadPath = null;

  /** Returns the current D2WContext. */
  public D2WContext d2wContext() {
    return d2wContext;
```

```
  }
  /** Sets the current D2WContext. */
  public void setLocalContext(D2WContext d2w) {
    d2wContext = d2w;
  }

  /** Returns the current EOEnterpriseObject. */
  public EOEnterpriseObject object() {
    return object;
  }
  /** Sets the current EOEnterpriseObject. */
  public void setObject(EOEnterpriseObject newEO) {
    object = newEO;
  }

  /** Creates an Excel Spreadsheet using Apache's POI project. If the
objectArray
    * is null or empty, this method will fetch all objects from the
database for
    * the given entityNameForArchive. If the  entityNameForArchive is null
or
    * empty, this method will fetch all entities and iterate over them,
while
    * fetching all objects for each entityName. This method will use aKey
to set
    * an Integer into the target, allowing for feedback while the
calculation is
    * occurring. The resulting file can be found in the /excel/ directory
of the
    * WebServer. */
  public String calculate(NSArray objectArray, String entityNameForArchive,
String aKey, WOComponent target) {

    try {
      EOEditingContext ec;
      HSSFWorkbook wb;
      NSArray entities;
      java.util.Enumeration entityEnum, attributeEnum, eoEnum;
      int sCount;
      String eName;
      HSSFSheet sheet;
      NSArray attributes;
      short col;
      HSSFRow row;
      String attribute, attribute2, value;
      short r;
      EOEnterpriseObject item;

      // set up the d2wcontext
      if (d2wContext() == null) setLocalContext( new
D2WContext(target.session()) );
      // set the task for archive
      d2wContext().setTask("archive");
```

```
      // get an EditingContext
      ec = target.session().defaultEditingContext();
      // create a workbook
      wb = new HSSFWorkbook();
      // get the entities
      if ((entityNameForArchive == null) || (entityNameForArchive.length()
== 0)) {
        entities = D2WUtils.allEntities();
        objectArray = null; // if there's no entity name, we'd better clear
out any objects.
      }
      else {
        entities = new
NSArray(EOUtilities.entityNamed(ec,entityNameForArchive));
      }
      // create an enumerator
      entityEnum = entities.objectEnumerator();
      // count the sheets
      sCount = 0;
      // enumerate
      while (entityEnum.hasMoreElements()) {
      // get the next entity
      eName = ((EOEntity)entityEnum.nextElement()).name();
      // set the entity into the D2WContext
      d2wContext().setEntity(EOUtilities.entityNamed(ec, eName));
      // get all objects for this entity
      if ((objectArray == null) || (objectArray.count() == 0)) objectArray
= EOUtilities.objectsForEntityNamed(ec, eName);
      // check if there are any objects
      if ((objectArray == null) || (objectArray.count() == 0)) continue;
      // create a worksheet
      sheet = wb.createSheet(displayName());
              // get the propertyKeys for this entity
      attributes = (NSArray)
d2wContext().valueForKey("displayPropertyKeys");
      // Label the columns
      attributeEnum = attributes.objectEnumerator();
      // zero out the column count
      col = 0;
              // Create a row and put some cells in it. Rows are 0 based.
              row = sheet.createRow((short)0);
      // iterate the attributes
      while ( attributeEnum.hasMoreElements() ) {
          // get the next attribute
          attribute = (String) attributeEnum.nextElement();
          // set the property into the D2WContext
          d2wContext().setPropertyKey(attribute);
          // insert it at the appropriate column
          // add the cell to the worksheet

row.createCell(col).setCellValue(d2wContext().displayNameForProperty());
          // move to the next column
          col++;
```

```
      }
      // Insert the Rows
      // make sure we start at row 1, otherwise we'll lose the column
headings
      r = 1;
      // set up to iterate the EOs
      eoEnum = objectArray.objectEnumerator();
      // iterate the EOs
      while ( eoEnum.hasMoreElements() ) {
          try {
          // zero out the column count
          col = 0;
          // create a new row
          row = sheet.createRow(r);
          // an EO
          item = (EOEnterpriseObject) eoEnum.nextElement();
          // set it into the D2W page
          setObject(item);
          // set up to iterate the attributes again...
          attributeEnum = attributes.objectEnumerator();
          // iterate the attributes again...
          while ( attributeEnum.hasMoreElements() ) {
              // the attribute
              attribute2 = (String) attributeEnum.nextElement();
              // set the property into the D2WContext
              d2wContext().setPropertyKey(attribute2);
              try {
              // the final value to insert from the current object based on
the d2wcontext
              value = currentValue();
              // create the cell for the spreadsheet
              row.createCell(col).setCellValue(value);
              }
              catch (Exception e1) {
              System.err.println(e1);
              }
              // move over a column
              col++;
          }
          }
          catch (Exception e2) {
          System.err.println(e2);
          }
          // move down a row, since there's a new object
          r++;
          target.takeValueForKey(new Integer(r),aKey);
      }
      // add another worksheet to the spreadsheet, since there's another
entity
      sCount++;
      // clear out the objectArray, since there's another entity
      objectArray = null;
```

```
      // System.out.println("POIArchive: calculate: Completed Archive: " +
eName);
      }
      // All sheets and cells added. Now write out the workbook
      String fn = fileUploadPath() + File.separator + "POI" + new
NSTimestamp().getTime() + ".xls";
      File file = new File(fn);
          FileOutputStream fileOut = new FileOutputStream(file);
          wb.write(fileOut);
          fileOut.close();

      objectArray = null;
      entityNameForArchive = null;

      System.out.println("POI @ " + fn);
      return fn;
  }
  catch (Exception e) {
      ByteArrayOutputStream baos = new ByteArrayOutputStream();
      e.printStackTrace(new PrintStream(baos));
      System.out.println("POIArchive: archive(): error:" +
baos.toString());
      return null;
  }
  }

  /** Returns the current value provided by the d2wContext. Found objects
use
    * the d2wContext for timestamp formatting, relationships, etc. If a
    * relationship fo an EOEnterpriseObject is resolved, and no
keyForRelationship
    * is found, this method will use the EO's userPresentableDescription.
*/
  public String currentValue() {
  Object o = null;
  try {
      if (d2wContext().propertyKeyIsKeyPath()) {
      o = NSKeyValueCodingAdditions.Utility.valueForKeyPath(object(),
d2wContext().propertyKey());
      } else {
      o = NSKeyValueCoding.DefaultImplementation.valueForKey(object(),
d2wContext().propertyKey());
      }
      if ( (o != null) && (o instanceof EOEnterpriseObject) ) {
      o = ((EOEnterpriseObject)
o).valueForKey(d2wContext().keyWhenRelationship());
      if ( (o != null) && (o instanceof EOEnterpriseObject) ) {
          try {
          return ((EOEnterpriseObject)o).userPresentableDescription();
          }
          catch (Exception ex) {
          ByteArrayOutputStream baos = new ByteArrayOutputStream();
          ex.printStackTrace(new PrintStream(baos));
```

```
          System.out.println("POIArchive: currentValue(): error" +
baos.toString());
          return o.toString();
          }
      }
      }
      if ( (o != null) && (o instanceof NSArray) ) {
      /***** ENHANCE ME: could use componentsSeparatedByString along with
userPresentableDescription. jpaul. *****/
      int count;
      count = ((NSArray) o).count();
      if (count == 0) return "0";
      return " " + count;
      }
      if ( (o != null) && (o instanceof NSTimestamp) ) {
      NSTimestampFormatter formatter = new NSTimestampFormatter((String)
d2wContext().valueForKey(D2WModel.FormatterKey));
      return formatter.format((NSTimestamp) o);
      }
      if ( (o != null) && (o instanceof java.lang.Number) ) {
      NSNumberFormatter formatter;
      String pattern = (String)
d2wContext().valueForKey(D2WModel.FormatterKey);
      if ((pattern != null) && (pattern.length() > 0)) {
          formatter = new NSNumberFormatter(pattern);
      }
      else {
          formatter = new NSNumberFormatter();
      }
      return formatter.stringForObjectValue(o);
      }
      if (o != null) return o.toString();
      return "";
  }
  catch (Exception e) {
      return "(Binary Data)";
  }
  }

  /** Returns an EOEnterpriseObject's 'displayNameForEntity', using the
d2wContext.
    * If it is not found, the method will check the userInfoDictionary for
    * displayName. If all else fails, it will return the entity's name. */
  public String displayName() {
  String s;

  s = (String) d2wContext().valueForKey("displayNameForEntity");
  if (s == null) {
      s = (String)
d2wContext().entity().userInfo().objectForKey("displayName");
      if (s != null) return s;
      return (String) d2wContext().entity().name();
  }
```

```
  return s;
  }

      /**
  * Returns the directory where uploaded files will be written to disk.
   */
  public static String fileUploadPath() {
  if (_fileUploadPath == null) {
      String tmpdir = System.getProperty("OTCUploadDirectory");
      if (tmpdir != null) {
      File tmpPath = new File(tmpdir);
      if (tmpPath.exists()) {
          _fileUploadPath = tmpPath.getAbsolutePath();
      }
      }
      if (_fileUploadPath == null) {
      NSLog.err.appendln("FileUpload: 'OTCUploadDirectory' does not exist."
          + " Please launch this application again with the
'OTCUploadDirectory'"
          + " System Property set to a directory to which you have write
permission.");
      }
  }
  return _fileUploadPath;
```

```
    }
  }
```