HI Shong
Thanks for the quick reply
For the First Error
I am sorry but I dint understand/find the importing of class part.
In the advance settings I don't see any thing where I can "Import the class".
I can do that in the tJava component but I use that component for testing output from a Component. So in the production version I will not use tJava component.
Attached the image for tFileList components advance settings.
Any example would be really helpfull.

Second Error Code

```
// ============================================================================
//
// Copyright (c) 2005-2008, Talend Inc.
//
// This source code has been automatically generated by Talend Open Studio
// / JobDesigner (CodeGenerator version 3.0.0.M3_r17341).
// You can find more information about Talend products at www.talend.com.
// You may distribute this code under the terms of the GNU LGPL license
// ( http://www.gnu.org/licenses/lgpl.html).
//
// ============================================================================
package weekly_dashboard.loadsitelevel_0_1;
import routines.DataOperation;
import routines.Mathematical;
import routines.Numeric;
import routines.Relational;
import routines.StringHandling;
import routines.TalendDataGenerator;
import routines.TalendDate;
import routines.TalendString;
import routines.system.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;
import java.math.BigDecimal;
import java.io.ByteArrayOutputStream;
import java.io.ByteArrayInputStream;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.io.IOException;
import java.util.Comparator;
//the import part of tJava_2
//import java.util.List;
/**
* Job: loadSiteLevel Purpose: For Loading Site Level Data<br>
* Description: This Job will pull data from Netezza Server and Pushes the data
* into the SQL server and Mysql datamart and check for new queues <br>
*
* @author surya.vikas@corp.aol.com
* @version 3.0.0.M3_r17341
```

```
* @status DEV
*/
public class loadSiteLevel {
// create and load default properties
private static java.util.Properties defaultProps = new java.util.Properties();
// create application properties with default
private static class ContextProperties extends java.util.Properties {
public ContextProperties(java.util.Properties properties) {
super(properties);
}
public ContextProperties() {
super();
}
}
private static ContextProperties context = new ContextProperties();
private static final String jobName = "loadSiteLevel";
private static final String projectName = "weekly_dashboard";
public static Integer errorCode = null;
private static String currentComponent = "";
private static final java.util.Map<String, Long> start_Hash = new java.util.HashMap<String, Long>();
private static final java.util.Map<String, Long> end_Hash = new java.util.HashMap<String, Long>();
private static final java.util.Map<String, Boolean> ok_Hash = new java.util.HashMap<String, Boolean>();
private static final java.util.Map<String, Object> globalMap = new java.util.HashMap<String, Object>();
public static final java.util.List<String[]> globalBuffer = new java.util.ArrayList<String[]>();
StatCatcherUtils tStatCatcher_1 = new StatCatcherUtils(
"_xYyeIISaEd2hKdL-4HMVRw", "0.1");
private class TalendException extends Exception {
private Exception e = null;
private loadSiteLevel c = null;
private TalendException(loadSiteLevel c, Exception e) {
this.e = e;
this.c = c;
}
@Override
public void printStackTrace() {
if (!(e instanceof TalendException || e instanceof TDieException)) {
globalMap.put(currentComponent + "_ERROR_MESSAGE", e
.getMessage());
System.err
.println("Exception in component " + currentComponent);
}
if (!(e instanceof TDieException)) {
e.printStackTrace();
}
if (!(e instanceof TalendException)) {
try {
for (java.lang.reflect.Method m : this.getClass()
.getEnclosingClass().getMethods()) {
if (m.getName().compareTo(currentComponent + "_error") == 0) {
m.invoke(c, new Object[] { e });
break;
}
}
```

```
if (!(e instanceof TDieException)) {
}
} catch (java.lang.SecurityException e) {
this.e.printStackTrace();
} catch (java.lang.IllegalArgumentException e) {
this.e.printStackTrace();
} catch (java.lang.IllegalAccessException e) {
this.e.printStackTrace();
} catch (java.lang.reflect.InvocationTargetException e) {
this.e.printStackTrace();
}
}
}
}
public void tFileList_1_error(Exception exception) throws TalendException {
end_Hash.put("tFileList_1", System.currentTimeMillis());
tStatCatcher_1.addMessage("failure", "tFileList_1", end_Hash
.get("tFileList_1")
- start_Hash.get("tFileList_1"));
tStatCatcher_1Process(globalMap);
tFileList_1_onSubJobError(exception);
}
public void tStatCatcher_1_error(Exception exception)
throws TalendException {
end_Hash.put("tStatCatcher_1", System.currentTimeMillis());
tStatCatcher_1_onSubJobError(exception);
}
public void tFileOutputDelimited_1_error(Exception exception)
throws TalendException {
end_Hash.put("tFileOutputDelimited_1", System.currentTimeMillis());
tStatCatcher_1_onSubJobError(exception);
}
public void tFileInputDelimited_1_error(Exception exception)
throws TalendException {
end_Hash.put("tFileInputDelimited_1", System.currentTimeMillis());
tFileList_1_onSubJobError(exception);
}
public void tJava_2_error(Exception exception) throws TalendException {
end_Hash.put("tJava_2", System.currentTimeMillis());
tFileList_1_onSubJobError(exception);
}
public void tJDBCConnection_1_error(Exception exception)
throws TalendException {
end_Hash.put("tJDBCConnection_1", System.currentTimeMillis());
tStatCatcher_1.addMessage("failure", "tJDBCConnection_1", end_Hash
.get("tJDBCConnection_1")
- start_Hash.get("tJDBCConnection_1"));
tStatCatcher_1Process(globalMap);
tJDBCConnection_1_onSubJobError(exception);
}
public void tFileList_1_onSubJobError(Exception exception)
throws TalendException {
try {
```

```
errorCode = null;
tStatCatcher_1Process(globalMap);
status = "end";
} catch (Exception e) {
e.printStackTrace();
}
}
public void tStatCatcher_1_onSubJobError(Exception exception)
throws TalendException {
}
public void tJDBCConnection_1_onSubJobError(Exception exception)
throws TalendException {
}
static class row2Struct implements
routines.system.IPersistableRow<row2Struct> {
final static byte[] commonByteArrayLock = new byte;
static byte[] commonByteArray = new byte;
String query;
public void readData(ObjectInputStream dis) {
synchronized (commonByteArrayLock) {
try {
int length = 0;
length = dis.readInt();
if (length == -1) {
this.query = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.query = new String(commonByteArray, 0, length);
}
} catch (IOException e) {
throw new RuntimeException(e);
}
}
}
public void writeData(ObjectOutputStream dos) {
try {
// String
if (this.query == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.query.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
} catch (IOException e) {
throw new RuntimeException(e);
```

```
}
}
public String toString() {
StringBuilder sb = new StringBuilder();
sb.append(super.toString());
sb.append("");
return sb.toString();
}
/**
* Compare keys
*/
public int compareTo(row2Struct other) {
int returnValue = -1;
return returnValue;
}
private int checkNullsAndCompare(Object object1, Object object2) {
int returnValue = 0;
if (object1 instanceof Comparable && object2 instanceof Comparable) {
returnValue = ((Comparable) object1).compareTo(object2);
} else if (object1 != null && object2 != null) {
returnValue = compareStrings(object1.toString(), object2
.toString());
} else if (object1 == null && object2 != null) {
returnValue = 1;
} else if (object1 != null && object2 == null) {
returnValue = -1;
} else {
returnValue = 0;
}
return returnValue;
}
private int compareStrings(String string1, String string2) {
// if (this.ignoreCase) {
return string1.compareToIgnoreCase(string2);
// } else {
// return string1.compareTo(string2);
// }
}
}
public void tFileList_1Process(final java.util.Map<String, Object> globalMap)
throws TalendException {
globalMap.put("tFileList_1_SUBPROCESS_STATE", 0);
try {
row2Struct row2 = new row2Struct();
/**
* start
*/
int NB_ITERATE_tFileInputDelimited_1 = 0; // for statistics
ok_Hash.put("tFileList_1", false);
start_Hash.put("tFileList_1", System.currentTimeMillis());
tStatCatcher_1.addMessage("begin", "tFileList_1");
tStatCatcher_1Process(globalMap);
currentComponent = "tFileList_1";
```

```
// tFileList_Begin
String directory_tFileList_1 = "C:/Queries";
String filemask_tFileList_1 = "*.txt";
filemask_tFileList_1 = org.apache.oro.text.GlobCompiler
.globToPerl5(filemask_tFileList_1.toCharArray(),
org.apache.oro.text.GlobCompiler.DEFAULT_MASK);
boolean case_sensitive_tFileList_1 = false;
java.util.regex.Pattern fileNamePattern_tFileList_1 = java.util.regex.Pattern
.compile(filemask_tFileList_1);
if (!case_sensitive_tFileList_1) {
fileNamePattern_tFileList_1 = java.util.regex.Pattern.compile(
filemask_tFileList_1,
java.util.regex.Pattern.CASE_INSENSITIVE);
}
java.io.File file_tFileList_1 = new java.io.File(
directory_tFileList_1);
final java.util.List<java.io.File> list_tFileList_1 = new java.util.ArrayList<java.io.File>();
file_tFileList_1.listFiles(new java.io.FilenameFilter() {
public boolean accept(java.io.File dir, String name) {
java.io.File file = new java.io.File(dir, name);
if (!file.isDirectory()) {
list_tFileList_1.add(file);
}
return true;
}
});
int NB_FILEtFileList_1 = 0;
for (int i_tFileList_1 = 0; i_tFileList_1 < list_tFileList_1.size(); i_tFileList_1++) {
java.io.File files_tFileList_1 = list_tFileList_1
.get(i_tFileList_1);
String fileName_tFileList_1 = files_tFileList_1.getName();
if (!fileNamePattern_tFileList_1.matcher(fileName_tFileList_1)
.matches()) {
continue;
}
String currentFileName_tFileList_1 = files_tFileList_1
.getName();
String currentFilePath_tFileList_1 = files_tFileList_1
.getAbsolutePath();
NB_FILEtFileList_1++;
globalMap.put("tFileList_1_CURRENT_FILE",
currentFileName_tFileList_1);
globalMap.put("tFileList_1_CURRENT_FILEPATH",
currentFilePath_tFileList_1);
/**
* stop
*/
/**
* start
*/
currentComponent = "tFileList_1";
/**
* stop
```

```
*/
NB_ITERATE_tFileInputDelimited_1++;
/**
* start
*/
ok_Hash.put("tJava_2", false);
start_Hash.put("tJava_2", System.currentTimeMillis());
currentComponent = "tJava_2";
String foo = "bar";
System.out.println(fid_((Integer) globalMap
.get("tFileInputDelimited_1_NB_LINE")));
/**
* stop
*/
/**
* start
*/
ok_Hash.put("tFileInputDelimited_1", false);
start_Hash.put("tFileInputDelimited_1", System
.currentTimeMillis());
currentComponent = "tFileInputDelimited_1";
org.talend.fileprocess.FileInputDelimited fid_tFileInputDelimited_1 = new org.talend.fileprocess.FileInputDelimited(
((String) globalMap.get("tFileList_1_CURRENT_FILEPATH"))
+ "\\"
+ ((String) globalMap
.get("tFileList_1_CURRENT_FILE")),
"ISO-8859-15", ";", "\n", true, 0, 0, -1, -1, false);
while (fid_tFileInputDelimited_1.nextRecord()) {
row2 = null;
boolean whetherReject_tFileInputDelimited_1 = false;
row2 = new row2Struct();
try {
row2.query = fid_tFileInputDelimited_1.get(0);
} catch (Exception e) {
whetherReject_tFileInputDelimited_1 = true;
System.err.println(e.getMessage());
row2 = null;
}
/**
* stop
*/
/**
* start
*/
currentComponent = "tFileInputDelimited_1";
/**
* stop
*/
// Start of branch "row2"
if (row2 != null) {
/**
* start
*/
```

```
currentComponent = "tJava_2";
/**
* stop
*/
} // End of branch "row2"
/**
* start
*/
currentComponent = "tFileInputDelimited_1";
}
fid_tFileInputDelimited_1.close();
globalMap.put("tFileInputDelimited_1_NB_LINE",
fid_tFileInputDelimited_1.getRowNumber());
ok_Hash.put("tFileInputDelimited_1", true);
end_Hash.put("tFileInputDelimited_1", System
.currentTimeMillis());
/**
* stop
*/
/**
* start
*/
currentComponent = "tJava_2";
ok_Hash.put("tJava_2", true);
end_Hash.put("tJava_2", System.currentTimeMillis());
/**
* stop
*/
/**
* start
*/
currentComponent = "tFileList_1";
}
globalMap.put("tFileList_1_NB_FILE", NB_FILEtFileList_1);
if (NB_FILEtFileList_1 == 0)
throw new RuntimeException("No file found in directory "
+ directory_tFileList_1);
ok_Hash.put("tFileList_1", true);
end_Hash.put("tFileList_1", System.currentTimeMillis());
tStatCatcher_1.addMessage("end", "tFileList_1", end_Hash
.get("tFileList_1")
- start_Hash.get("tFileList_1"));
tStatCatcher_1Process(globalMap);
/**
* stop
*/
} catch (Exception e) {
throw new TalendException(this, e);
}
globalMap.put("tFileList_1_SUBPROCESS_STATE", 1);
}
static class row1Struct implements
routines.system.IPersistableRow<row1Struct> {
```

```
final static byte[] commonByteArrayLock = new byte;
static byte[] commonByteArray = new byte;
java.util.Date moment;
String pid;
String father_pid;
String root_pid;
Long system_pid;
String project;
String job;
String job_repository_id;
String job_version;
String context;
String origin;
String message_type;
String message;
Long duration;
public void readData(ObjectInputStream dis) {
synchronized (commonByteArrayLock) {
try {
int length = 0;
length = dis.readByte();
if (length == -1) {
this.moment = null;
} else {
this.moment = new Date(dis.readLong());
}
length = dis.readInt();
if (length == -1) {
this.pid = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.pid = new String(commonByteArray, 0, length);
}
length = dis.readInt();
if (length == -1) {
this.father_pid = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.father_pid = new String(commonByteArray, 0, length);
```

```
}
length = dis.readInt();
if (length == -1) {
this.root_pid = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.root_pid = new String(commonByteArray, 0, length);
}
length = dis.readByte();
if (length == -1) {
this.system_pid = null;
} else {
this.system_pid = dis.readLong();
}
length = dis.readInt();
if (length == -1) {
this.project = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.project = new String(commonByteArray, 0, length);
}
length = dis.readInt();
if (length == -1) {
this.job = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.job = new String(commonByteArray, 0, length);
}
length = dis.readInt();
if (length == -1) {
this.job_repository_id = null;
} else {
```

```
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.job_repository_id = new String(commonByteArray, 0,
length);
}
length = dis.readInt();
if (length == -1) {
this.job_version = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.job_version = new String(commonByteArray, 0,
length);
}
length = dis.readInt();
if (length == -1) {
this.context = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.context = new String(commonByteArray, 0, length);
}
length = dis.readInt();
if (length == -1) {
this.origin = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.origin = new String(commonByteArray, 0, length);
```

```
}
length = dis.readInt();
if (length == -1) {
this.message_type = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.message_type = new String(commonByteArray, 0,
length);
}
length = dis.readInt();
if (length == -1) {
this.message = null;
} else {
if (length > commonByteArray.length) {
if (length < 1024 && commonByteArray.length == 0) {
commonByteArray = new byte;
} else {
commonByteArray = new byte;
}
}
dis.readFully(commonByteArray, 0, length);
this.message = new String(commonByteArray, 0, length);
}
length = dis.readByte();
if (length == -1) {
this.duration = null;
} else {
this.duration = dis.readLong();
}
} catch (IOException e) {
throw new RuntimeException(e);
}
}
}
public void writeData(ObjectOutputStream dos) {
try {
// java.util.Date
if (this.moment == null) {
dos.writeByte(-1);
} else {
dos.writeByte(0);
dos.writeLong(this.moment.getTime());
}
// String
if (this.pid == null) {
dos.writeInt(-1);
```

```
} else {
byte[] byteArray = this.pid.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.father_pid == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.father_pid.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.root_pid == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.root_pid.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// Long
if (this.system_pid == null) {
dos.writeByte(-1);
} else {
dos.writeByte(0);
dos.writeLong(this.system_pid);
}
// String
if (this.project == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.project.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.job == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.job.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.job_repository_id == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.job_repository_id.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
```

```
if (this.job_version == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.job_version.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.context == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.context.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.origin == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.origin.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.message_type == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.message_type.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// String
if (this.message == null) {
dos.writeInt(-1);
} else {
byte[] byteArray = this.message.getBytes();
dos.writeInt(byteArray.length);
dos.write(byteArray);
}
// Long
if (this.duration == null) {
dos.writeByte(-1);
} else {
dos.writeByte(0);
dos.writeLong(this.duration);
}
} catch (IOException e) {
throw new RuntimeException(e);
}
}
public String toString() {
StringBuilder sb = new StringBuilder();
sb.append(super.toString());
```

```
sb.append("");
return sb.toString();
}
/**
* Compare keys
*/
public int compareTo(row1Struct other) {
int returnValue = -1;
return returnValue;
}
private int checkNullsAndCompare(Object object1, Object object2) {
int returnValue = 0;
if (object1 instanceof Comparable && object2 instanceof Comparable) {
returnValue = ((Comparable) object1).compareTo(object2);
} else if (object1 != null && object2 != null) {
returnValue = compareStrings(object1.toString(), object2
.toString());
} else if (object1 == null && object2 != null) {
returnValue = 1;
} else if (object1 != null && object2 == null) {
returnValue = -1;
} else {
returnValue = 0;
}
return returnValue;
}
private int compareStrings(String string1, String string2) {
// if (this.ignoreCase) {
return string1.compareToIgnoreCase(string2);
// } else {
// return string1.compareTo(string2);
// }
}
}
public void tStatCatcher_1Process(
final java.util.Map<String, Object> globalMap)
throws TalendException {
globalMap.put("tStatCatcher_1_SUBPROCESS_STATE", 0);
try {
row1Struct row1 = new row1Struct();
/**
* start
*/
ok_Hash.put("tFileOutputDelimited_1", false);
start_Hash
.put("tFileOutputDelimited_1", System.currentTimeMillis());
currentComponent = "tFileOutputDelimited_1";
String fileName_tFileOutputDelimited_1 = (new java.io.File(
"C:/ETL/TalendNew/TOS-All-r17341-V3.0.0M3/workspace/WEEKLY_DASHBOARD/logs/waste.log"))
.getAbsolutePath().replace("\\", "/");
String fullName_tFileOutputDelimited_1 = null;
String extension_tFileOutputDelimited_1 = null;
String directory_tFileOutputDelimited_1 = null;
```

```
if ((fileName_tFileOutputDelimited_1.indexOf("/") != -1)) {
if (fileName_tFileOutputDelimited_1.lastIndexOf(".") < fileName_tFileOutputDelimited_1
.lastIndexOf("/")) {
fullName_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1;
extension_tFileOutputDelimited_1 = "";
} else {
fullName_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1
.substring(0, fileName_tFileOutputDelimited_1
.lastIndexOf("."));
extension_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1
.substring(fileName_tFileOutputDelimited_1
.lastIndexOf("."));
}
directory_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1
.substring(0, fileName_tFileOutputDelimited_1
.lastIndexOf("/"));
} else {
if (fileName_tFileOutputDelimited_1.lastIndexOf(".") != -1) {
fullName_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1
.substring(0, fileName_tFileOutputDelimited_1
.lastIndexOf("."));
extension_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1
.substring(fileName_tFileOutputDelimited_1
.lastIndexOf("."));
} else {
fullName_tFileOutputDelimited_1 = fileName_tFileOutputDelimited_1;
extension_tFileOutputDelimited_1 = "";
}
directory_tFileOutputDelimited_1 = "";
}
int nb_line_tFileOutputDelimited_1 = 0;
int splitEvery_tFileOutputDelimited_1 = 1000;
int splitedFileNo_tFileOutputDelimited_1 = 0;
int currentRow_tFileOutputDelimited_1 = 0;
final String OUT_DELIM_tFileOutputDelimited_1 = /**
* Start field
* tFileOutputDelimited_1:FIELDSEPARATOR
*/
";"/** End field tFileOutputDelimited_1:FIELDSEPARATOR */
;
final String OUT_DELIM_ROWSEP_tFileOutputDelimited_1 = /**
* Start field
* tFileOutputDelimited_1:ROWSEPARATOR
*/
"\n"/** End field tFileOutputDelimited_1:ROWSEPARATOR */
;
// create directory only if not exists
if (directory_tFileOutputDelimited_1 != null
&& directory_tFileOutputDelimited_1.trim().length() != 0) {
java.io.File dir_tFileOutputDelimited_1 = new java.io.File(
directory_tFileOutputDelimited_1);
if (!dir_tFileOutputDelimited_1.exists()) {
dir_tFileOutputDelimited_1.mkdirs();
```

```
}
}
// routines.system.Row
java.io.Writer outtFileOutputDelimited_1 = new java.io.BufferedWriter(
new java.io.OutputStreamWriter(
new java.io.FileOutputStream(
fileName_tFileOutputDelimited_1, false),
"ISO-8859-15"));
java.io.File filetFileOutputDelimited_1 = new java.io.File(
fileName_tFileOutputDelimited_1);
/**
* stop
*/
/**
* start
*/
ok_Hash.put("tStatCatcher_1", false);
start_Hash.put("tStatCatcher_1", System.currentTimeMillis());
currentComponent = "tStatCatcher_1";
for (StatCatcherUtils.StatCatcherMessage scm : tStatCatcher_1
.getMessages()) {
row1.pid = pid;
row1.root_pid = rootPid;
row1.father_pid = fatherPid;
row1.project = projectName;
row1.job = jobName;
row1.context = contextStr;
row1.origin = (scm.getOrigin() == null
|| scm.getOrigin().length() < 1 ? null : scm
.getOrigin());
row1.message = scm.getMessage();
row1.duration = scm.getDuration();
row1.moment = scm.getMoment();
row1.message_type = scm.getMessageType();
row1.job_version = scm.getJobVersion();
row1.job_repository_id = scm.getJobId();
row1.system_pid = scm.getSystemPid();
/**
* stop
*/
/**
* start
*/
currentComponent = "tStatCatcher_1";
/**
* stop
*/
/**
* start
*/
currentComponent = "tFileOutputDelimited_1";
StringBuilder sb_tFileOutputDelimited_1 = new StringBuilder();
if (row1.moment != null) {
```

```
sb_tFileOutputDelimited_1.append(
FormatterUtils.format_Date(row1.moment,
"yyyy-MM-dd HH:mm:ss")
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.pid != null) {
sb_tFileOutputDelimited_1.append(
row1.pid
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.father_pid != null) {
sb_tFileOutputDelimited_1.append(
row1.father_pid
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.root_pid != null) {
sb_tFileOutputDelimited_1.append(
row1.root_pid
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.system_pid != null) {
sb_tFileOutputDelimited_1.append(
row1.system_pid
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.project != null) {
sb_tFileOutputDelimited_1.append(
row1.project
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.job != null) {
sb_tFileOutputDelimited_1.append(
row1.job
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.job_repository_id != null) {
sb_tFileOutputDelimited_1.append(
row1.job_repository_id
);
```

```
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.job_version != null) {
sb_tFileOutputDelimited_1.append(
row1.job_version
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.context != null) {
sb_tFileOutputDelimited_1.append(
row1.context
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.origin != null) {
sb_tFileOutputDelimited_1.append(
row1.origin
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.message_type != null) {
sb_tFileOutputDelimited_1.append(
row1.message_type
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.message != null) {
sb_tFileOutputDelimited_1.append(
row1.message
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_tFileOutputDelimited_1);
if (row1.duration != null) {
sb_tFileOutputDelimited_1.append(
row1.duration
);
}
sb_tFileOutputDelimited_1
.append(OUT_DELIM_ROWSEP_tFileOutputDelimited_1);
outtFileOutputDelimited_1.write(sb_tFileOutputDelimited_1
.toString());
nb_line_tFileOutputDelimited_1++;
/**
* stop
*/
/**
* start
```

```
*/
currentComponent = "tStatCatcher_1";
}
ok_Hash.put("tStatCatcher_1", true);
end_Hash.put("tStatCatcher_1", System.currentTimeMillis());
/**
* stop
*/
/**
* start
*/
currentComponent = "tFileOutputDelimited_1";
outtFileOutputDelimited_1.close();
globalMap.put("tFileOutputDelimited_1_NB_LINE",
nb_line_tFileOutputDelimited_1);
ok_Hash.put("tFileOutputDelimited_1", true);
end_Hash.put("tFileOutputDelimited_1", System.currentTimeMillis());
/**
* stop
*/
} catch (Exception e) {
throw new TalendException(this, e);
}
globalMap.put("tStatCatcher_1_SUBPROCESS_STATE", 1);
}
public void tJDBCConnection_1Process(
final java.util.Map<String, Object> globalMap)
throws TalendException {
globalMap.put("tJDBCConnection_1_SUBPROCESS_STATE", 0);
try {
/**
* start
*/
ok_Hash.put("tJDBCConnection_1", false);
start_Hash.put("tJDBCConnection_1", System.currentTimeMillis());
tStatCatcher_1.addMessage("begin", "tJDBCConnection_1");
tStatCatcher_1Process(globalMap);
currentComponent = "tJDBCConnection_1";
java.lang.Class.forName("org.netezza.Driver");
String connString_tJDBCConnection_1 = "jdbc:netezza://NZ-D02.db.aol.com:5480/nzdw1rpt";
String userName_tJDBCConnection_1 = "suryavikas";
String password_tJDBCConnection_1 = "Vikas007";
java.sql.Connection conn_tJDBCConnection_1 = java.sql.DriverManager
.getConnection(connString_tJDBCConnection_1,
userName_tJDBCConnection_1,
password_tJDBCConnection_1);
conn_tJDBCConnection_1.setAutoCommit(false);
globalMap.put("conn_tJDBCConnection_1", conn_tJDBCConnection_1);
globalMap
.put("url_tJDBCConnection_1", connString_tJDBCConnection_1);
// globalMap.put("user_tJDBCConnection_1",
// userName_tJDBCConnection_1);
// globalMap.put("pass_tJDBCConnection_1",
```

```
// password_tJDBCConnection_1);
/**
* stop
*/
/**
* start
*/
currentComponent = "tJDBCConnection_1";
/**
* stop
*/
/**
* start
*/
currentComponent = "tJDBCConnection_1";
ok_Hash.put("tJDBCConnection_1", true);
end_Hash.put("tJDBCConnection_1", System.currentTimeMillis());
tStatCatcher_1.addMessage("end", "tJDBCConnection_1", end_Hash
.get("tJDBCConnection_1")
- start_Hash.get("tJDBCConnection_1"));
tStatCatcher_1Process(globalMap);
/**
* stop
*/
} catch (Exception e) {
throw new TalendException(this, e);
}
globalMap.put("tJDBCConnection_1_SUBPROCESS_STATE", 1);
}
public static boolean watch = false;
public static int portStats = 3334;
public static int portTraces = 4334;
public static String clientHost;
public static String defaultClientHost = "localhost";
public static String contextStr = "Default";
public static String pid = "0";
public static String rootPid = null;
public static String fatherPid = null;
public static long startTime = 0;
private static ThreadLocal threadLocal = new ThreadLocal();
private static java.util.Properties context_param = new java.util.Properties();
public static String status = "";
public static void main(String[] args) {
int exitCode = runJobInTOS(args);
globalBuffer.clear();
System.exit(exitCode);
}
public static String[][] runJob(String[] args) {
int exitCode = runJobInTOS(args);
String[][] bufferValue = new String[][] { { Integer.toString(exitCode) } };
globalBuffer.clear();
return bufferValue;
}
```

```
public static synchronized int runJobInTOS(String[] args) {
init();
String lastStr = "";
for (String arg : args) {
if (arg.equalsIgnoreCase("--context_param")) {
lastStr = arg;
} else if (lastStr.equals("")) {
evalParam(arg);
} else {
evalParam(lastStr + " " + arg);
lastStr = "";
}
}
if (clientHost == null) {
clientHost = defaultClientHost;
}
pid = TalendString.getAsciiRandomString(6);
if (rootPid == null) {
rootPid = pid;
}
if (fatherPid == null) {
fatherPid = pid;
}
try {
java.io.InputStream inContext = loadSiteLevel.class
.getClassLoader()
.getResourceAsStream(
"weekly_dashboard/loadsitelevel_0_1/contexts/Default.properties");
if (inContext != null) {
defaultProps.load(inContext);
inContext.close();
context = new ContextProperties(defaultProps);
}
if (contextStr.compareTo("Default") != 0) {
inContext = loadSiteLevel.class.getClassLoader()
.getResourceAsStream(
"weekly_dashboard/loadsitelevel_0_1/contexts/"
+ contextStr + ".properties");
if (inContext != null) {
context.load(inContext);
inContext.close();
}
}
if (!context_param.isEmpty()) {
context.putAll(context_param);
}
} catch (java.io.IOException ie) {
System.err.println("Could not load context " + contextStr);
ie.printStackTrace();
}
long startUsedMemory = Runtime.getRuntime().totalMemory()
- Runtime.getRuntime().freeMemory();
long endUsedMemory = 0;
```

```
long end = 0;
startTime = System.currentTimeMillis();
final loadSiteLevel loadSiteLevelClass = new loadSiteLevel();
loadSiteLevelClass.tStatCatcher_1.addMessage("begin");
try {
loadSiteLevelClass.tStatCatcher_1Process(globalMap);
} catch (Exception e) {
e.printStackTrace();
}
try {
errorCode = null;
loadSiteLevelClass.tFileList_1Process(globalMap);
status = "end";
} catch (TalendException e_tFileList_1) {
status = "failure";
e_tFileList_1.printStackTrace();
globalMap.put("tFileList_1_SUBPROCESS_STATE", -1);
} finally {
}
try {
errorCode = null;
loadSiteLevelClass.tJDBCConnection_1Process(globalMap);
status = "end";
} catch (TalendException e_tJDBCConnection_1) {
status = "failure";
e_tJDBCConnection_1.printStackTrace();
globalMap.put("tJDBCConnection_1_SUBPROCESS_STATE", -1);
} finally {
}
end = System.currentTimeMillis();
if (watch) {
System.out.println((end - startTime) + " milliseconds");
}
endUsedMemory = Runtime.getRuntime().totalMemory()
- Runtime.getRuntime().freeMemory();
if (false) {
System.out.println((endUsedMemory - startUsedMemory)
+ " bytes memory increase when running : loadSiteLevel");
}
loadSiteLevelClass.tStatCatcher_1.addMessage(status == "" ? "end"
: status, (end - startTime));
try {
loadSiteLevelClass.tStatCatcher_1Process(globalMap);
} catch (Exception e) {
e.printStackTrace();
}
reset();
if (errorCode == null) {
return status != null && status.equals("failure") ? 1 : 0;
} else {
return errorCode.intValue();
}
}
```

```
public static void evalParam(String arg) {
if (arg.startsWith("--watch")) {
watch = true;
} else if (arg.startsWith("--stat_port=")) {
portStats = Integer.parseInt(arg.substring(12));
} else if (arg.startsWith("--trace_port=")) {
portTraces = Integer.parseInt(arg.substring(13));
} else if (arg.startsWith("--client_host=")) {
clientHost = arg.substring(14);
} else if (arg.startsWith("--context=")) {
contextStr = arg.substring(10);
} else if (arg.startsWith("--father_pid=")) {
fatherPid = arg.substring(13);
} else if (arg.startsWith("--root_pid=")) {
rootPid = arg.substring(11);
} else if (arg.startsWith("--context_param")) {
String keyValue = arg.substring(16);
int index = -1;
if (keyValue != null && (index = keyValue.indexOf('=')) > -1) {
context_param.put(keyValue.substring(0, index), keyValue
.substring(index + 1));
}
}
}
private static void init() {
errorCode = null;
status = "";
}
private static void reset() {
defaultProps.clear();
context.clear();
threadLocal = new ThreadLocal();
currentComponent = "";
start_Hash.clear();
end_Hash.clear();
ok_Hash.clear();
globalMap.clear();
watch = false;
portStats = 3334;
portTraces = 4334;
clientHost = null;
defaultClientHost = "localhost";
contextStr = "Default";
pid = "0";
rootPid = null;
fatherPid = null;
context_param.clear();
System.gc();
}
}
/************************************************************************************************
* 49723 characters generated by Talend OpenStudio on the September 25, 2008
* 2:53:39 PM IST
```

**********/

**********ERROR **********
Starting job loadSiteLevel at 14:54 25/09/2008.
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
The method fid_(Integer) is undefined for the type loadSiteLevel
org.talend.fileprocess cannot be resolved to a type
org.talend.fileprocess cannot be resolved to a type
at weekly_dashboard.loadsitelevel_0_1.loadSiteLevel.tFileList_1Process(loadSiteLevel.java:391)
at weekly_dashboard.loadsitelevel_0_1.loadSiteLevel.runJobInTOS(loadSiteLevel.java:1599)
at weekly_dashboard.loadsitelevel_0_1.loadSiteLevel.main(loadSiteLevel.java:1507)
Job loadSiteLevel ended at 14:54 25/09/2008.

Thanks in Advance
Surya