ContribTableExample

NOTE: This is outdated information that only applies to Tapestry 4.1 and earlier.

This is a simple implementation of contrib:table. for those of you looking for example code:

```
How to implement tapestry contrib:Table
=======================================

HTML part
---------

::

  <table jwcid="table@contrib:Table"
         source="ognl:model"
         columns="ognl:columnModel"
         pageSize="50"
         class="style-normal"
         pagesClass="style-normal"
         columnsClass="style-normal"
         rowsClass="style-normal"
         valuesClass="style-normal">
  </table>

Java component part
-------------------

::

  public abstract class Logs extends BasePage implements PageBeginRenderListener {

    //getter and setters for data and column model:
    public abstract IBasicTableModel getModel();
    public abstract void setModel(IBasicTableModel model);
    public abstract ITableColumnModel getColumnModel();
    public abstract void setColumnModel(ITableColumnModel columnModel);

    //persist model for request scope
    public void pageBeginRender(final PageEvent event) {
      final IBasicTableModel newModel = createDataModel();
      final ITableColumnModel columnModel = getLogsLogic().createColumnModel();

      setModel(newModel);
      setColumnModel(columnModel);
    }

    private IBasicTableModel createDataModel() {
      return new IBasicTableModel() {
        private List list;
        private long count = -1;

        public Iterator getCurrentPageRows(final int offset,
                                           final int pageSize,
                                           final ITableColumn iTableColumn,
                                           final boolean orderDirection) {
          if (null == list) {
            //retrieve data on a row basis from business logic
            list = getLogsLogic().retrieveDbData(offset, pageSize, iTableColumn, orderDirection);
          }
          return list.iterator();
        }

        public int getRowCount() {
          if (-1 == count) {
            count = getLogsLogic().retrieveDbDataCount();
          }
          return (int) count;
```

```
        }

      };
    }

    //map business logic service
    public abstract LogsLogic getLogsLogic();

  }
```

Java business logic part

```
::

  public List retrieveDbData(int offset,
                             int pageSize,
                             ITableColumn orderColumn,
                             boolean ascending) {
    String queryString = "select \n" +
            "    t.id, \n" +                                    //0
            "    t.targetIdAsString, \n" +                      //1
            "    c.id, \n" +                                    //2
            "    ts.id, \n" +                                   //3
            "    ts.type, \n" +                                 //4
            "    ts.status, \n" +                               //5
            "    ts.startOccurrence, \n" +                      //6
            "    ts.endOccurrence, \n" +                        //7
            "    tss.id, \n" +                                  //8
            "    tss.type, \n" +                                //9
            "    tss.occurrence, \n" +                          //10
            "    tss.message\n" +                               //11
            "from \n" +
            "    Target t, \n" +
            "    ChainRun c, \n" +
            "    Task ts, \n" +
            "    TaskStep tss  \n" +
            "where t.id = c.targetId \n" +
            "and c.id = ts.chainRunId \n" +
            "and ts.id = tss.taskId";

    if (null != orderColumn) {
      queryString += " order by " + orderColumn.getColumnName() + (ascending ? " ASC" : " DESC");
    }

    org.hibernate.classic.Session session = HibernateTool.getSessionFactory().getCurrentSession();
    session.beginTransaction();

    Query query = session.createQuery(queryString);

    query.setFirstResult(offset);
    query.setMaxResults(pageSize);

    List list = query.list();
    return list;
  }

  public long retrieveDbDataCount(){
    String queryString = "select count(*)\n" +
            "from \n" +
            "    Target t, \n" +
            "    ChainRun c, \n" +
            "    Task ts, \n" +
            "    TaskStep tss  \n" +
            "where t.id = c.targetId \n" +
            "and c.id = ts.chainRunId \n" +
            "and ts.id = tss.taskId";
    org.hibernate.classic.Session session = HibernateTool.getSessionFactory().getCurrentSession();
    session.beginTransaction();
    Query query = session.createQuery(queryString);
    Long count = (Long) query.uniqueResult();
```

```
  return count;
}

public ITableColumnModel createColumnModel() {
  return new ITableColumnModel() {
    //use a sorted set to have the columns sorted by key...
    private SortedMap<Integer, ITableColumn> mapping = new TreeMap<Integer, ITableColumn>();

    {
      ITableColumnEvaluator columnEvaluator = new ITableColumnEvaluator() {
        public Object getColumnValue(ITableColumn iTableColumn, Object row) {
          Object[] dataSet = (Object[]) row;

          for (Map.Entry<Integer, ITableColumn> e : mapping.entrySet()) {
            Integer idx = e.getKey();
            ITableColumn column = e.getValue();
            if (column.getColumnName().equals(iTableColumn.getColumnName())) {
              Object value = dataSet[idx];
              if (value instanceof Date)
                return DateUtil.dateTimeFormatCH.format(value);
              else
                return value;
            }
          }
          return null;
        }
      };

      mapping.put(1, new SimpleTableColumn("t.targetIdAsString", "LIID", columnEvaluator, true));
      mapping.put(2, new SimpleTableColumn("c.id", "Chain Run Id", columnEvaluator, true));
      mapping.put(4, new SimpleTableColumn("ts.type", "Task Type", columnEvaluator, true));
      mapping.put(5, new SimpleTableColumn("ts.status", "Status", columnEvaluator, true));
      mapping.put(9, new SimpleTableColumn("tss.type", "Type", columnEvaluator, true));
      mapping.put(10, new SimpleTableColumn("tss.occurrence", "Occurrence", columnEvaluator, true));
      mapping.put(11, new SimpleTableColumn("tss.message", "Message", columnEvaluator, true));
    }

    public ITableColumn getColumn(final String s) {
      for (ITableColumn iTableColumn : mapping.values()) {
        if (iTableColumn.getColumnName().equals(s))
          return iTableColumn;
      }
      return null;
    }

    public int getColumnCount() {
      return mapping.size();
    }

    public Iterator<ITableColumn> getColumns() {
      return mapping.values().iterator();
    }
  };
}
```